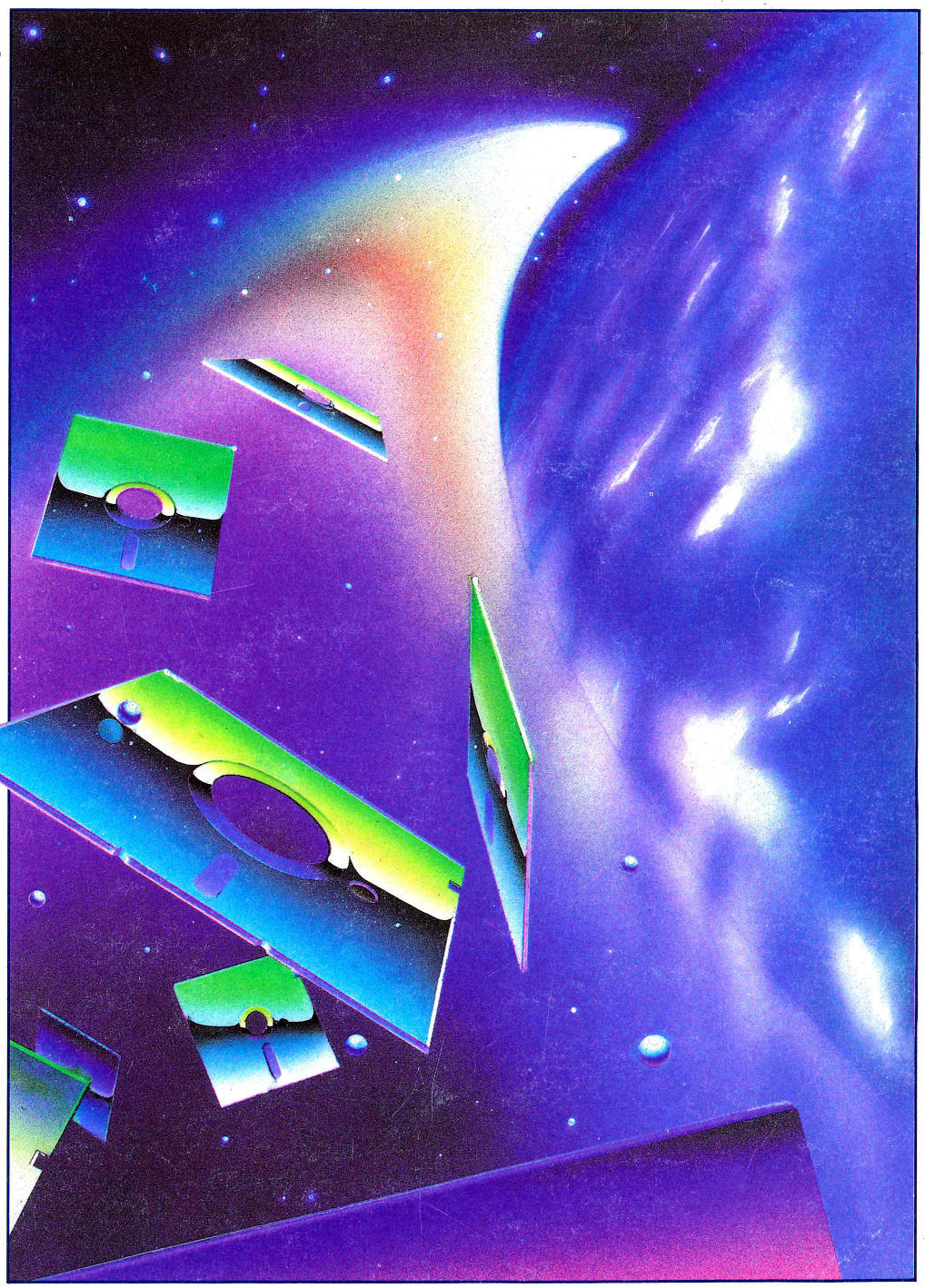
AN ASHTON-TATE SOFTWARE SUPPORT PUBLICATION

MARCH 1986

TechNotes

- Understanding Views in dBASE III PLUS
- The IIF() Function
- Generating Menus in PC/MS-DOS
- Variable Tab Stops in Framework II
- Cumulative Index January through March 1986



TechNotes

TABLE OF CONTENTS

FEATURES

4 Understanding Views
by Nicole Hannes

8 The IIF() Function
by Christopher White

12 SET COLOR from a Menu
by Stephen Goodman

14 Automating Mailing Labels
by Lisa Wanek

18 Variable Tab Stops
by Quinn Wildman

21 Printing Program Listings
by Jack Hirsch

25 Exchanging Numbers and Text in a Cell
by Gail Kulikowsky

27 Using Rundbase on a Hard Disk
by Clay Carley

29 Generating Menus in PC/MS-DOS
by Stephen Goodman

36 Converting PFS Files to Framework II
by Quinn Wildman

DEPARTMENTS

1 Read Me First
by Laurie Miller

3 Corrigenda

39 Usage Tips

44 Letters

50 Update Policies

57 Index

READ ME FIRST

by Laurie Miller

Welcome to another issue of ***TechNotes***! This month we are featuring three articles that pertain to dBASE III PLUS, the newest product in the dBASE product line. With the introduction of dBASE III PLUS comes, among numerous other features, the ability to create and use a "view." Nicole Hannes's article, "Understanding Views," offers an insightful discussion of how views can allow easier access to data stored in related database files.

Another article that is of interest to dBASE III PLUS users is Christopher White's "The IIF() Function." Chris talks about how you can use this handy new function in conjunction with expressions in REPORT FORMs, LABEL FORMs, and INDEXing, a capability that was previously not available with these commands.

The third article in this issue featuring dBASE III PLUS programming is Stephen Goodman's "SET COLOR from a Menu." His program saves the COLOR settings to HISTORY for later use and allows an interactive color definition. (Another menu-related article by Steve, "Generating Menus in PC/MS-DOS," makes interesting reading for those of you who want a fast, simple, and inexpensive front-end menu system.)

A Framework II article featured this month is "Automating Mailing Labels," by Lisa Wanek. Although Framework II has improved upon Framework 1.x's capability for generating mailing labels, Lisa's article presents a program that automates the entire process, offering the user flexibility in what the finished product looks like.

Demonstrating the increased flexibility of word processing capabilities in Framework II is "Variable Tab Stops," by Quinn Wildman. Discussed is a FRED utility program which allows you to redefine the **Tab** key.

I have received many letters from dBASE II users who would like to see more technical information related to dBASE II covered in ***TechNotes***. We are always happy and willing to accommodate the interests of our readers, but we don't know what our dBASE II users'needs are. If there is a technical problem, question, program, idea, or whatever, that is unique to dBASE II that you would like to see addressed and/or published in ***TechNotes***, please send it to me c/o Ashton-Tate Software Support.

Again in this month's issue is the Reader Survey. I encourage all of our readers to take about five minutes to fill this out and return it to us in the postage-paid business reply envelope. We want to know what it is you like and dislike about ***TechNotes*** so that we can adjust it to your needs if necessary. Thank you for your input and continued readership.

Publisher
Ashton-Tate Software Support Center
Jay Hanson, Director
Roy Moore, Manager

Editor-in-Chief
Laurie Miller

Editors
Kent Irwin, Christopher White

Assistant Editor
Nicole Hannes

Technical Staff
Stanley Ballenger, Jeffrey Bersin, Olivier Biggerstaff, Robert Boies, Paul Bukauskas, Richard Bunter, Clay Carley, Fred Crownover, Michael Fiore, Richard Goepel, Stephen Goodman, Doug Haman, Jack Hirsch, David Howlett, Errol Jackson, Matt Johnson, Rosaline Keenan, Gail Kulikowsky, Steven Kurasch, Chris Lovasco, Elizabeth Miranda, Joseph Perret, John Regen, Karen Robinson, Steve Silverwood, Lisa Wanek, Quinn Wildman, Tom Williams, Leonard Zerman

Administrative Staff
Carol Jackson, April Larson, Kathy Smith

Illustrator
Stephen Goodman

Cover Artist
Darrell Evers

Layout Design
Dusty Lee, D. A. Gray Bare

Typesetting
Renée Maynard, Carol Heys

Index
Nicole Hannes

Product Development
Forefront Development Center:
Marilyn McMahon, Greg Stikeleather
Glendale Development Center:
Debby Moody

DISCLAIMER NOTICE

The information contained in this publication is designed to help you develop and improve your programming skills and to assist you in using Ashton-Tate products. Ashton-Tate assumes no responsibility whatsoever for the uses made of any of the software code in this issue, whether modified or unmodified.

If you discover what you believe is an anomaly (as defined under Anomalies) not presented in this issue or others, please send a detailed description of the possible anomaly to:

Ashton-Tate Software Support Center
20101 Hamilton Ave.
Torrance, CA 90502-1319

If the reported anomaly can be reproduced by our technicians, it may be added to a future issue of ***TechNotes***.

The policies that are described in ***TechNotes*** apply only to the United States (and possessions), Canada, and Mexico. Residents of other foreign countries should contact a local Ashton-Tate office for policy information and technical support.

Ashton-Tate and the authors will not be liable for special, incidental, consequential, indirect or other similar damages, even if we have been advised of the possibility of such damages. This means we are not responsible for damages or costs incurred as a result of loss of time, loss of data, loss of profits or revenues, or loss of use of the software. In addition, we are not responsible for damages or costs incurred in connection with obtaining substitute software, claims by others, inconvenience or similar costs.

No warranties, either expressed or implied, are made regarding the contents of this publication, its merchantability or fitness for any particular purpose, all of which are hereby expressly disclaimed. Note, however, some states do not allow the exclusion or limitation of warranties or of direct, incidental or consequential damages, so these warranty limitations may not apply to you.

In no event will our liability for any damages to you or any other person ever exceed the price paid for your subscription to ***TechNotes***, regardless of any form of the claim.

Additional statements by agents, employees, distributors and dealers of Ashton-Tate do not constitute warranties by Ashton-Tate and do not bind Ashton-Tate.

TechNotes (ISSN 0883-6493) is an Ashton-Tate Software Support Center publication containing technical reference information about three of Ashton-Tate's products: dBASE II®, dBASE III PLUS®, and Framework II™.

Entire contents Copyright © Ashton-Tate 1986. All rights reserved. Commercial reproduction of the material appearing in ***TechNotes*** is forbidden without prior written permission. Direct all requests for reproduction permission to Ashton-Tate Software Support Center, 20101 Hamilton Ave., Torrance, CA 90502-1319 ATTN: Laurie Miller.

TechNotes is published monthly at a subscription price of $50.00 for one year. Additional rates: Canada and Mexico, $65.00; other countries, $75.00. Monthly issues are available in back issues for $10.00 each (1985 issues) and $7.00 each (1986 issues). ***TechNotes*** is published by the Ashton-Tate Software Support Center, 20101 Hamilton Ave., Torrance, CA 90502-1319. POSTMASTER: Send subscription inquiries and address changes to ***TechNotes*** Subscription Service, P.O. Box 3729, Escondido, CA 92025-0929.

dBASE II and dBASE III are registered trademarks of Ashton-Tate. dBASE, RunTime, FRED, Framework, and ASHTON-TATE are trademarks of Ashton-Tate. Some of the trademarks of other companies mentioned in this issue appear for identification purposes only.

Corrigenda

1. The 1985 Annual ***TechNotes*** Disk that was included with the December 1985 issue of ***TechNotes*** is formatted for the IBM PC and 100% compatibles. If you are running under an operating system other than PC/MS-DOS, contact a downloading service to obtain the appropriate format. (Usually a local users group will know about downloading services in the area.) We apologize for not clearly stating the disk's format directly on the label.

2. The routine for generating subtotals from a database in the July 1985 issue of ***TechNotes***, pages 54 through 59, has a small anomaly. The current way it appears will not properly calculate the final totals. The flaw is in the 9 line statement which begins on the bottom of page 56 and ends on the top of page 57.

   The current listing is as follows:

```
Report := Report &
          "***** Totals *****" & @rept(" ", 26 ) &
          @rept( " ", 9 - @len( @decimal( total1 ) ) ) &
          @decimal( total1 ) &
          @rept( " ", 9 - @len( @decimal( total2 ) ) ) &
          @decimal( total2 ) &
          @rept( " ", 9 - @len( @decimal( total3 ) ) ) &
          @decimal( total3 ) &
          newline,
```

   The listing should be:

```
Report := Report &
          "***** Totals *****" & @rept(" ", 26 ) &
          @rept( " ", 9 - @len( @decimal( total1 + subtotal1 ) ) ) &
          @decimal( tota1 + subtotal1 ) &
          @rept( " ", 9 - @len( @decimal( total2 + subtotal2 ) ) ) &
          @decimal( total2 + subtotal2 ) &
          @rept( " ", 9 - @len( @decimal( total3 + subtotal3 ) ) ) &
          @decimal( total3 + subtotal3 ) &
          newline,
```

3. Ralph Davis's article, "Creating a Field List and EDITing Selected Fields," on page 6 in the December 1985 issue, failed to mention that the program (Fldlist.PRG) was intended to run under the Developer's Release of dBASE III.

4. There is an error on page 19 of the December 1985 issue in "SED—The dBASE Screen Editor," by Karen Robinson. The paragraph that begins: "If your system is configured with the CP/M-86 operating system SED can be run from more than one subdirectory," is incorrect. CP/M does not support subdirectories. Please disregard this paragraph.

   In the same issue and article, on page 22, Karen Robinson offers a work-around of using spaces instead of the **Tab** key to indent. This will not work correctly because SED creates a hard **Tab** for every eight spaces anyway. (See "Letters," in this issue, for a work-around.)

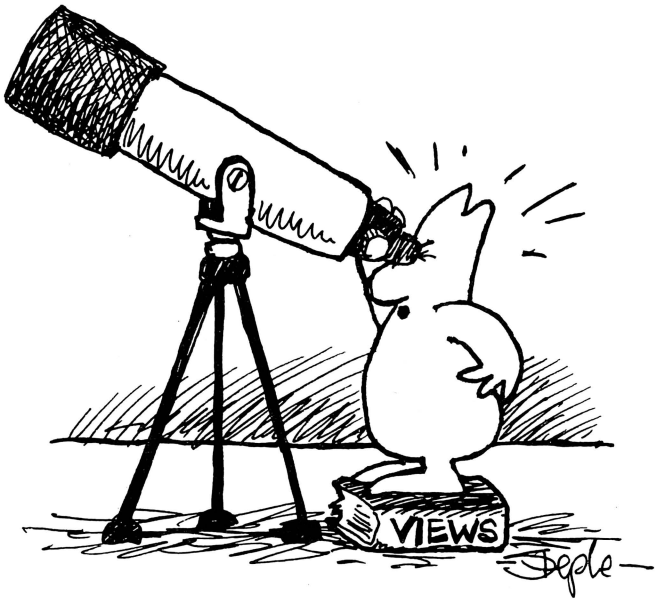
5. The dBASE II anomaly item reported on page 27 of the October 1985 issue ("Sending @...SAYs to the Printer") is incorrect. The problem as it appears was correctly reported on page 33 of the September 1985 issue ("SET MARGIN").

Understanding Views

by Nicole Hannes

Views are among dBASE III PLUS's new features that enhance the relational abilities of dBASE. A view is a virtual database file, a window that allows you to look into and access selected information from several database files as if they are a single database file. Views are conceptual or logical—not physical—database files.

The view's primary purpose is to define, model, and maintain subsets and groupings of database files as a logical data structure. There are several advantages to working with a logical data structure in this way instead of an actual file. First, you can concentrate only on the data that concerns you and ignore the rest. Second, you can create virtual JOINs of several database files and work with them quickly thereby increasing your database modeling and testing abilities, without using the JOIN or other commands. Third, users can view the same database files in different ways, which is especially useful when referring to reports and labels. Last, the physical database file structure can be made transparent to the user, allowing the physical structures to change while retaining the logical data structures the end-user manipulates.



In dBASE III PLUS, you can construct a view from related database files, associated index files, a format file, and a filter. Using SET FIELDS, you select only the fields you need to access from database files constituting the view. Then, when you execute commands that support the FIELDS clause, such as AVERAGE, BROWSE, CHANGE, COUNT, EDIT, LIST, or TOTAL, only the specified fields are operational. If you have created a view containing several database files that are linked with the appropriate SET RELATION commands, only the records that have matching keys along the relational chain will constitute the view's record contents.

An important point to understand is that views do not contain data themselves, but are represented and maintained in definition files, called view files, which have the (.VUE) extension. There are two ways to create these special files: (1) use the full-screen mode CREATE/MODIFY VIEW, or (2)record the current settings for files and fields with CREATE VIEW <filename> FROM ENVIRONMENT.

The command SET VIEW TO <(.VUE) filename> brings a view into use. All defined files are opened, RELATIONs SET, FILTERs SET, AND FIELDs SET. You can then examine and update your data.

Views with One Database File

The simplest use of a view is to create a subset of a single database file. This subset can consist of fields, records, or both. For example, let's say you have a database file, Customers.DBF, that has the following structure and is INDEXed ON Name.

Field	Field Name	Type	Width	Dec
1	Name	Character	20	
2	Address	Character	25	
3	City	Character	20	
4	State	Character	2	
5	Zip	Character	5	
6	Open_Date	Date	8	
7	Last_Trans	Date	8	
8	Last_Amt	Numeric	7	2
9	Balance	Numeric	7	2

Suppose that you wish to narrow the fields list to contain only Name, Address, City, State, and Zip for simple mailing list queries. The following commands CREATE the desired VIEW:

```
USE Customers INDEX Name
SET FIELDS TO Name, Address, City, State, Zip
CREATE VIEW Mail FROM ENVIRONMENT
```

Once you have CREATEd the (.VUE) file, the commands to set up and LIST your mailing list become:

```
SET VIEW TO Mail
LIST
```

To extend the example further, you can narrow the scope of your subset by restricting the database file to a set of records that meet a specific condition. Suppose, you now want to print labels for only those records where the state is California. In addition, since this is a regular task, you want to maintain this subset of fields and records as a view. This is how to set up your view:

```
USE Customers INDEX Name
SET FIELDS TO Name, Address, City, State, Zip
SET FILTER TO State = "CA"
CREATE VIEW Calif FROM ENVIRONMENT
```

When you are ready to print labels for the state of California using a predefined LABEL FORM, use the commands:

```
SET VIEW TO Calif
LABEL FORM <label filename> TO PRINT
```

If you have a format file to EDIT the customer names, the following steps would be necessary:

```
USE Customers INDEX Name
SET FORMAT TO Name
EDIT FIELDS Name, Address, City, State, Zip
```

By utilizing the view file, Customers.VUE, the steps become:

```
SET VIEW TO Customers
EDIT
```

Here, the view opens the database file with the specified INDEX, and then SETs FORMAT TO the appropriate (.FMT) file.

dBASE III PLUS allows you to define several views for each database file, each view accessing different forms of your data. Using views in this way also means that several users can access the same data in a form that is individually tailored for each user.

Views with Multiple Database Files

Views become very useful when you are working with more than one database file. Often you need to relate two or more database files and access certain fields from both files. For example, suppose you have your customer list, Customer.DBF, and Orders.DBF, a database file of outstanding orders to fulfill. Now, suppose you need to phone all the customers with orders outstanding for more than 30 days. To extract this information, SET a RELATION between the two database files, then SET a FILTER for accessing only those records that correspond to today's date minus 30.

There are two ways to accomplish this with views. First, using CREATE VIEW, you can easily SET the RELATION with menu options. Second, if you are working from the command line, you can CREATE VIEW <name> FROM ENVIRONMENT to save the relationship.

For example, the typical sequence of commands that define the view for the set of matching records:

```
USE Customer INDEX Name
SELECT 2
USE Orders INDEX Ord_Date
SET RELATION TO Name INTO Customer
SET FIELDS TO Ord_Date, Customer->Name, Customer->Phone
SET FILTER TO Ord_Date < DATE() - 30
```

At this point, issue the command CREATE VIEW Overdue FROM ENVIRONMENT. Now, anytime you want to access the names and phone numbers of the customers with outstanding orders older than 30 days, issue the command SET VIEW TO Overdue. Any operation, such as DISPLAY, LIST, EDIT, or BROWSE, will allow you access to the specified fields and records.

Notice that creating this type of view is equivalent to merging Orders.DBF and Customers.DBF with the JOIN command. However, the view can be invoked immediately and does not have to be explicitly maintained as the JOINed database file does. The view preserves the logical data independence of the outstanding orders list from the database files that constitute it. The JOIN file, on the other hand, either must be maintained or recreated every time you wish to work with data from both Orders.DBF and Customers.DBF.

SET FIELDS

SET FIELDS TO is a new dBASE III PLUS command that creates a global pool of fields accessible from any SELECTed work area as if they are all a part of one large database file. This is the major contributor to the relational and field scoping capabilities of views, and in combination with SET RELATION creates the virtual database file that a view represents. Read the discussion starting on page 5-169 of the Using section of the *dBASE III PLUS Reference Manual* to understand how some commands operate differently when there is a FIELD list active.

To reference a field in the pool, simply specify the name of the field without an ALIAS. The ALIAS will only need to be specified if there are fields in multiple work areas with the same name. Attempting to reference fields that are not in the fields pool after the SET FIELDS TO command has been executed and FIELDS are SET ON, will return the error message "Variable not found," if the command accessing the field supports the FIELDS list, such as LIST or CHANGE.

With SET FIELDS TO in dBASE III PLUS, you can access and EDIT fields from unSELECTed work areas in all full-screen modes with the except APPEND. This includes CHANGE, EDIT, REPLACE, and @...GET, as long as you have a RELATION SET between the database files. If there is no RELATION SET, only the fields from the current work area are available. In addition, the FIELDS list is supported in REPORT and LABEL FORMs. Press **F10** to access fields from the global fields list in the contents definition areas of the REPORT and LABEL FORM. All fields that display are those FIELDS currently SET.

The SET FIELDS TO <fields list> command works with SET FIELDS ON/OFF command. When you SET FIELDS TO <field list>, the SET FIELDS toggle is turned ON automatically. In turn when you open a new database file in an unSELECTed work area, the FIELDS toggle is SET OFF.

Some Notes About Views

When working with views there are some important points to keep in mind.

1. When you CREATE VIEW <filename> FROM ENVIRONMENT, the last SELECTed work area becomes the active work area when you use the view. In other words, the view will open all the database files, but SELECTs as current the work area that was active when you issued the CREATE VIEW <filename> FROM ENVIRONMENT command.
2. Format files are local to the work area in which they were created. Each database file can have its own format format. So the active format file for the view will be the format file that is in the active work area.

To summarize, views are virtual database files that are defined by and maintained in (.VUE) files. These (.VUE) files are created by the CREATE/MODIFY VIEW or CREATE VIEW FROM ENVIRONMENT commands. There are two basic forms of views. The first is a record or field subset of a single database file. This view is a row and/or column window into your database file and is constructed with SET FILTER and SET FIELDS. A more complex view is constructed from multiple database files. This view presents a window into several database files at one time and is constructed from SET RELATION, SET FIELDS, and SET FILTER. What makes this form of views most intriguing is that it represents a virtual JOIN of not merely two physical database files but several. Be sure to consult the Using section of the *dBASE III PLUS Reference Manual*, and refer to the command descriptions as you learn the new power of views.

The IIF() Function

by Christopher White

The IIF() function, first introduced in the Developer's Release of dBASE III, is one of the most important functions added to the dBASE language. It adds conditional capability within expressions so that the result of a logical expression returns a specified value depending on whether the result is true (.T.) or false (.F.). The value that is returned can be any data type except Memo. The function syntax is as follows:

IIF(<logical expression>, <true value>, <false value>)

The one requirement is that both the true and the false paths return a value of the same data type. You cannot, for example, have a numeric value returned if the result of the logical expression is true and a character value if the result is false. In this case, the IIF() function will return the error message "Invalid function argument."

As a programming tool the IIF() is very flexible, powerful, and can be used in a myriad of ways. For example, it can convert logical variables to variables of character, numeric, or date type. It can be used in any command that supports expressions, no matter what the input expression data type is, given the flexibility of the data type it can return. As an applications tool, the IIF() function is extraordinary for how it can enhance REPORT and LABEL FORM, @...SAY output, and INDEXing. If there is corner you can get stuck in, odds are that the IIF() function can work you out of it.

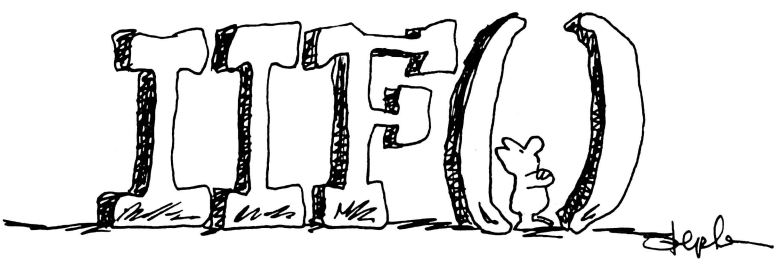
The discussion that follows demonstrates some of its uses from the applications point of view. Each section presents a small problem and a solution using the IIF() function. It is hoped that this discussion will lead you toward using this function in all the corners of your programming world.

REPORT FORM

Blank if Zero

Often in business reports you need to specify a numeric report field that displays an alternative string if the field value is zero. For example, if you want to display a dash character in place of a zero in a report, the IIF() expression that may be inserted into a REPORT FORM would be the following, assuming the numeric field is called Amount:

IIF(Amount = 0, " - ", STR(Amount))



Converting Logical Fields to Character Strings

Logical fields are quite useful for many operations. They make conditional statements easy to define, are easy to validate, and save file space. The major drawback when using them in REPORT FORMs, however, is the unsightly display of the logical data type. You can easily format logical fields with the IIF() function and create a REPORT FORM field display that is both appealing and meaningful. For example, suppose you have an inventory control system in which you have defined a logical field to flag an item as on-order when items are back ordered. When you report a list of items you would like a status field to display the phrase "Back Order" if the item has been ordered.

Inventory Report

Item Number	Order Date	Description	On Hand	Status
1212	02/01/86	Widget	0	Back Order
1213	02/05/86	Thing	12	

The REPORT FORM field expression would be defined as follows:

```
IIF( onorder, "Back Order", "" )
```

LABEL FORM

The IIF() function can be used in several ways in the LABEL FORM to assist the process of formatting the output.

One of the small problems that occurs when printing labels is how to deal with the period following a middle initial field if the field is blank. Prior to the IIF() function, if the period was specified and the middle initial field was blank, the period was standing alone between the first and last name fields. A solution, and a painful one, was to define the middle initial field as two characters then entering the period into the field if the person had a middle initial. With the IIF() function, this problem is simply and elegantly solved with the following expression:

```
IIF( Mi = " ","", Mi + "." )
```

Now when the middle initial is blank, the first and last name fields are separated by a single space otherwise the middle initial is followed by a period.

Another small, but persistent problem formatting LABEL FORMs is the inability to skip blank lines. The LABEL FORM, of course, suppresseś blank lines automatically. For some applications, this is not a blessing. Some labels, particularly shipping labels require a fixed format with embedded blank lines if the label field is blank. It is an unpleasant prospect to have to give up the ease of use the LABEL FORM generator provides and program multiple across fixed format labels with a command file. Again the IIF() function solves the problem with ease. Define an expression like the one specified below so that if the label field is blank, the IBM Extended character 255 is used, which has no display value. It now appears as if your label skips a line where the label field is blank.

```
IIF( "" = TRIM( Address ), CHR( 255 ), Address )
```

INDEXing

Logical Fields

The IIF() function can be used to convert variables of logical data type to values of character or numeric data type. This is particularly useful for defining index expressions that include logical fields. For example, suppose you have a transactions database file that is ordered by client's last name. In addition, you have a logical field that flags the bill status of the transaction. When you make a bill run you would like to start at the first non-billed transaction and then print bills until all open transactions are billed. To do this, you need to INDEX your transactions database file by bill status and client name, in that order. The problem is that the bill status field is logical. The following demonstrates how to use the IIF() to convert the logical field to a single character string in order to create an index file ordered by the logical field and concatenate the result with another character field in the subject database file.

```
* ---Create an index file that separates billed
* ---from nonbilled items.
INDEX ON IIF( Billed, "Y", "N" ) + Client TO Billed
* ---At bill time, find the first nonbilled item.
SEEK "N"
```

Conditional Keys

Another INDEXing problem that can be solved with the IIF() function is the creation of conditional keys. If, for example, you have a mailing list database file that contains records for both companies and individuals, you will want to INDEX on the name of the individual. But, if the name field or expression is blank, you want the key value to be the company name. The following example demonstrates what the list report should look like.

Client Address List

Last	First	Company
Arnold	Wandy	
		Rasta Man Properties
Shallow	Elizabeth	Rasta Man Properties

The INDEX command to create this order is:

```
INDEX ON IIF( "" = TRIM( Last + First ),;
Company, Last + First ) TO Condndx
```

@...SAY

A common occurrence with pre-defined forms, such as insurance forms in medical applications, is that some display fields are boxes. Sex is a field that often will display as an ''X'' or check mark in a box. For example,

```
          M  F
Sex       X
```

No matter how you have defined the sex field, the IIF() function can be used to determine the column in which the ''X'' is to display. The expression to put the ''X'' in the appropriate box is as follows:

```
@ 10, IIF( Sex = "M", 13, 17 ) SAY "X"
```

As has been demonstrated by the small number of problem examples, the IIF() is both useful and powerful. The IIF() is the definitive user function for all levels of applications.

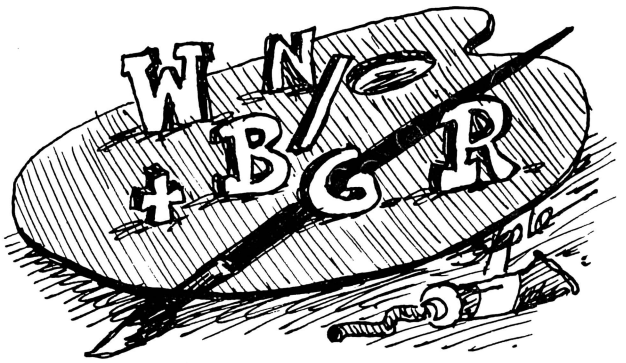
SET COLOR from a Menu

by Stephen Goodman

If you have a color monitor or are developing an application that will use color, you probably need a user-friendly program that will assist your user in setting color. The full-screen SET command does this but allows the user to configure several other parameters at the same time. The command file that follows, Color.PRG, presents an alternative. You can set colors from a full and interactive screen that appears very much like the new dBASE III PLUS full-screen modes. A menu is presented that offers choices of colors followed by three input boxes, one each for the color definitions of SAYs, GETs, and the screen border. Both foreground and background are prompted for. The color definition, as it will be executed in a SET COLOR TO command, is displayed on the status line with the definition codes stated as the letters. For example, the status line will look like this:

SET COLOR TO *W/N,N/W,N* Color definition codes.

The program is useful as an interactive color definition or can be integrated into applications either for interactive color definitions or installations.




```
* Program ...: Color.PRG
* Author ....: Stephen Goodman
* Date ......: March 1, 1986
* Note(s) ...: Program to SET COLOR from a menu in dBASE III PLUS.
*
SET TALK OFF
SET STATUS OFF
default = IIF( ISCOLOR(), "W+,GR+/B+" ,"W/N,N/W" )
ok = .N.
DO WHILE .NOT. ok
   SET COLOR TO &default
```

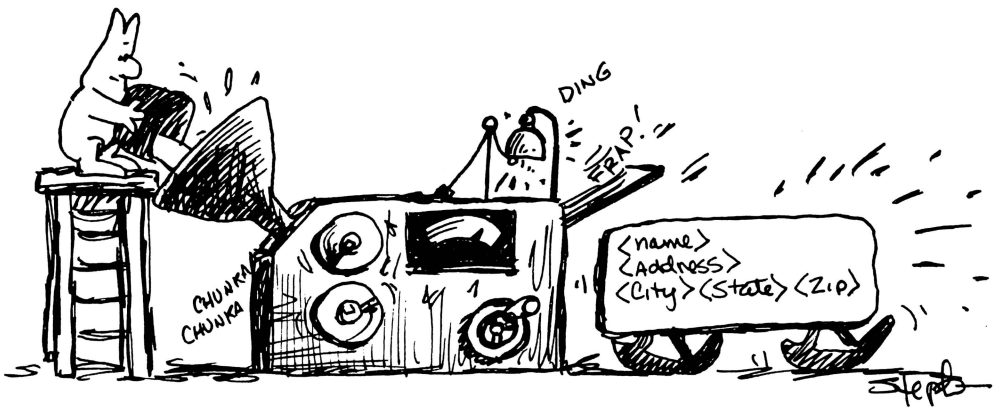
```
   CLEAR
   STORE 0 TO say1, say2, get1, get2, brdr
   ? " C h o i c e  o f  C o l o r s"
   ? REPLICATE(CHR(205),79)
   ?
   ? "   0 - Black      8 - Grey       "
   ? "   1 - Blue       9 - Hi Blue    "
   ? "   2 - Green     10 - Hi Green   "
   ? "   3 - Cyan      11 - Hi Cyan    "
   ? "   4 - Red       12 - Hi Red     "
   ? "   5 - Magenta   13 - Hi Magenta "
   ? "   6 - Brown     14 - Hi Yellow  "
   ? "   7 - White     15 - Hi White   "
   ?
   @ 14,5 SAY "SAY Colors"
   @ 15,3 TO 18,26
   @ 16,5  SAY "Enter foreground:" ;
           GET say1 PICTURE "99" RANGE 0,15
   @ 17,5  SAY "Enter background:" ;
           GET say2 PICTURE "99" RANGE 0,15
   @ 14,30 SAY "GET Colors"
   @ 15,28 TO 18,51
   @ 16,30 SAY "Enter foreground:" ;
           GET get1 PICTURE "99" RANGE 0,15
   @ 17,30 SAY "Enter background:" ;
           GET get2 PICTURE "99" RANGE 0,15
   @ 14,55 SAY "BORDER Color"
   @ 15,53 TO 18,76
   @ 16,55 SAY "Enter border: " ;
           GET brdr PICTURE "99" RANGE 0,15
   SET COLOR TO I
   @ 22,0 SAY "SET COLOR TO" + SPACE( 15 ) ;
              + CHR( 186 ) + SPACE( 25 ) ;
              + CHR( 186 ) + SPACE( 24 )
   SET COLOR TO &default
   @ 24,29 SAY "Enter color definitions"
   READ
   IF READKEY() = 12
      RETURN
   ENDIF
   colors = "N B G BG R BR GR W N+ B+ G+ BG+R+ BR+GR+W+"
   csay1 = LTRIM(TRIM(SUBSTR(colors,(say1*3)+1,3)))
   csay2 = LTRIM(TRIM(SUBSTR(colors,(say2*3)+1,3)))
   cget1 = LTRIM(TRIM(SUBSTR(colors,(get1*3)+1,3)))
   cget2 = LTRIM(TRIM(SUBSTR(colors,(get2*3)+1,3)))
   cbrdr = LTRIM(TRIM(SUBSTR(colors,(brdr*3)+1,3)))
   colorset = csay1+"/"+csay2+","+cget1+"/"+cget2+","+cbrdr
   SET COLOR TO I
   @ 22,13 SAY colorset
      SET COLOR TO &default
   @ 23,00 CLEAR
   @ 24,31 SAY "Is this OK (Y/N) " GET ok PICTURE "Y"
   READ
   IF ok
      SET COLOR TO &colorset
   ENDIF
ENDDO
SET TALK ON
RETURN
* EOP Color.PRG
```

Automating Mailing Labels

by Lisa Wanek

There are times when it helps to automate even the simplest task. Framework II has greatly facilitated the task of generating mailing labels, and the further automation of this process, especially if frequently used, can be very helpful. For instance, you may normally use just a few different formats of labels, so generating the template frames each time may be redundant.

The following program, Labelpgm, exemplifies the kind of automation you might want to implement. It consists of two major parts. The first part allows you to input all of the print formatting information. You are then left on your own to input the actual field names in the label frame using the < > brackets. The second part of the program copies the frames across according to the number of labels you wish to have on a row, then sets appropriate print settings, and prints the labels.



This program allows you to generate any size label and to choose from one to five labels across the page. You may wish to include only the most commonly used label types for your application. It may be easier to have existing templates for two or three types of labels, and allow the selection of the label type needed. Of course the best solution is the one that most suits your needs. To set up this program, create an empty word frame and label it Labelpgm. Then go inside this frame and create six more empty word frames. Label them Hi, Up, Wide, Database, Size, and Print. (Note: Database is an empty word frame, not a database.)

Now, go back to the border of Labelpgm. Press **EDIT FORMULA (F2)** and **ZOOM (F9)**. Then enter these lines of code:

```
; Labelpgm
; by Lisa Wanek

@setmacro({Alt-L},Labelpgm.Size),
@setmacro({Alt-C},Labelpgm.Print)
```

Press **Esc**, then **Dnlevel** into the frame. In the formula area of Size, enter:

```
; Size

; This program allows you to input all the information
; about the labels you wish to produce, and creates a label frame
; of the appropriate size. You then input the data to
; be printed inside the generated frame using the < > brackets.
; The Alt-C command will invoke the Print program which will copy
; across the correct number of labels and, using the designated
; database (on the desktop), print the labels.

@local(name,size),

; Input from user to determine label size
; and which database to print.

Up:=@inputline("Enter number of labels across(1-5)",,,,#YES),
Wide:=@inputline("Enter label width (# char across)(1-240)",,,,#YES),
Hi:=@inputline("Enter height of label (# lines)(1-16)",,,,#YES),
Database:=@inputline("Enter name of database to be used",,,,),
name:="ONE",
@performkeys("{Ctrl-Uplevel}{Ctrl-C}E{Return}Mainlabel{Return}" &
"{Dnlevel}{Ctrl-C}E{Return}"),
@performkeys("{Ctrl-F}A"),

; Name and size the first frame.

@performkeys(name &"{Return}{F4}{Ctrl-Home}{Return}"),
@if(@len(Wide)=2,
   @performkeys("{F4}{Ctrl-"&@mid(Wide,1,1)&
         "}{Ctrl-"&@mid(Wide,2,1)&"}{Rightarrow}"),
   @performkeys("{F4}{Ctrl-"&Wide&"}{Rightarrow}")
),
@if(@len(Hi)=2,
   @performkeys("{Ctrl-"&@mid(Hi,1,1)&
         "}{Ctrl-"&@mid(Hi,2,1)&"}{Dnarrow}{Return}"),
   @performkeys("{Ctrl-"&Hi&"}{Dnarrow}{Return}")
),
@performkeys("{Dnlevel}")
```

Press **Esc** when you are finished. Now go to the frame called Print. In its formula area, enter this formula:

```
; Print

; This program copies the frame across according to number of
; labels across specified and prints the labels specified.

@local(name,widelp,hilp,uplp,go),
@performkeys("{Uplevel}"),
```

```
; The following loop is designed to create the necessary labels
; and move them into the appropriate position for printing.
; Uplp is used as a counter.

uplp:=1,
@while(uplp<>@value(Up),
     widelp:=@value(Wide)+1,
     name:=@choose(uplp,"TWO","THREE","FOUR","FIVE"),

     ; Copy the current frame and put in drag mode.
     @performkeys("{F8}{Return}"),
     @performkeys("{F3}"&@rept("{Rightarrow}",widelp)),
     @performkeys("{Return}"&name&"{Return}"),
uplp:=uplp+1,
),

; Print labels.
@performkeys("{Ctrl-Home}{F6}{End}{Return}"),
@if(@sense(#FD),
     @performkeys("{Ctrl-F}D"),
     #NULL!
),
@if(@sense(#FH),
     #NULL!,
     @performkeys("{Ctrl-F}H")
),
@performkeys("{Uplevel}"),

; Set print attributes.
@performkeys("{Ctrl-P}FPL"&Hi&"{Return}OO{Return}"),
@performkeys("HTO{Return}FBO{Return}{Uplevel}{Uplevel}"),

; Choose print labels option from Apps
; menu and specify database for Printing.
go:="@performkeys(""{Ctrl-A}P"&Database&"{Return}"")",
@go
```

After you have the program all keyed in and ready to run, here are the steps to produce labels.

1. Load the program frame, Labelpgm, and your database onto the desktop.
2. With your cursor on the frame border of the Labelpgm frame, press **RECALC (F5)** to initialize the macros.
3. Press **Alt-L** to begin the program.
4. You will be asked to enter the following information about the labels you wish to print:
   - Height of label (at least one more than the actual number of lines you wish to print)
   - Width of label
   - Number of labels across
   - Name of database frame

5. The label frame will be created and sized accordingly.
6. You are then left inside the frame to enter the field names to be printed in < > brackets. For example:

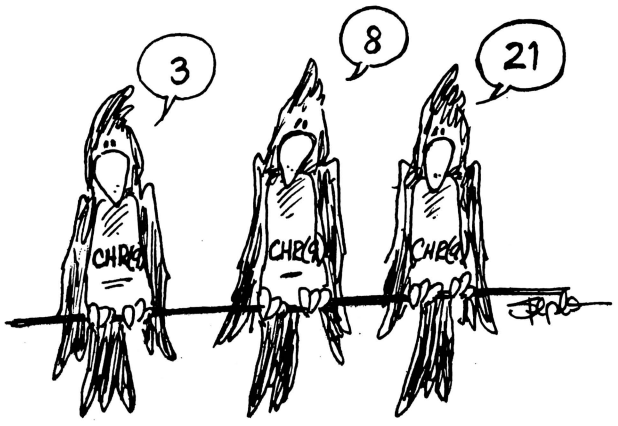
   <name>
   <address>
   <city>, <state> <zip>

7. When finished, type **Alt-C** from inside the template frame to continue the program.
8. The frame will then be copied across, all print attributes set, and the labels printed.

Variable Tab Stops

by Quinn Wildman

One advanced word processing feature that Framework II does not support is variable position tab stops. Normally the **Tab** key will move the cursor forward to the next character column that is evenly divisible by the tab size, plus one. If your tab size is 5 you may tab to positions 6, 11, 16 and so forth. But what if you want something else?



The following small program, Tabstop, implements variable tab stopping by redefining the **Tab** key with the **@keyfilter** function. The Tabstop program looks at the cursor position on the current line using **@panel1** and then inserts the number of spaces necessary to put the cursor at the next tab stop.

Here is how to set up Tabstop:

1. Create an empty word frame and label it Tabstop.
2. Press the **Dnlevel** key to enter Tabstop and create three empty word frames. Label them Definition, Do, and Quit. Next create a spreadsheet (still inside the Tabstop frame) with 2 columns and 10 rows. Label the spreadsheet Stops. (The Stops spreadsheet will be used to store the tab stop settings.)
3. Enter the following formulas into the formula areas of the indicated frames.

Tabstop:

```
; Variable Tab Stop Macro
; by Quinn Wildman
@setmacro({Alt-Q},Tabstop.Quit),
@keyfilter({Tab},Tabstop.Do),
@setmacro({Alt-S},Tabstop.Definition)
```

Tabstop.Definition:

```
@echo(#OFF),
@local(current,flag),
@fill(Stops.A1:Stops.A10,256),
flag := #TRUE,
@while(flag,
   current:=@inputline("Enter next tab stop, or press escape to exit."),
   @if(@key={Esc},
      flag:=#FALSE,
      @list(
         @put(Stops.A1:Stops.A10,@value(current)),
         @next(Stops.A1:Stops.A10),
         @if(@get(Stops.A1:Stops.A10)=#NULL!,
            @list(
               flag:=#FALSE,
               @inputline("No more than 10 tab stops allowed, "&
                  "press any key to continue")
            )
         )
      )
   )
),
@setselection("Tabstop.Stops"),
@performkeys("{Dnlevel}{Ctrl-Home}{F6}{Ctrl-Dnarrow}{Ctrl-L}A"),
@echo(#ON)
```

Tabstop.Do:

```
@local(spaces,temp),
spaces := 0,
@reset(Stops.A1:Stops.A10),
@while(@and(@panel1 >= @get(Stops.A1:Stops.A10),
            @get(Stops.A1:Stops.A10)<>#NULL!),
   @next(Stops.A1:Stops.A10)
),
@if(@get(Stops.A1:Stops.A10)=256,
   @list(
      @echo(#OFF),
      @performkeys("{Ctrl-W}R{F6}{End}{Return}"),
      temp := @value(@textselection),
      @performkeys("{Return}"),
      @echo(#ON),
      spaces := temp-@panel1+Stops.A1+1
   ),
   spaces:=@get(Stops.A1:Stops.A10)-@panel1
),
@performkeys(@rept(" ",spaces))
```

Tabstop.Quit:

```
@keyfilter({Tab})
```

4. Press **RECALC (F5)** on the Tabstop frame. Then Press **Alt-S** and enter up to ten tab stops by specifying the column number. Anytime you wish to redefine your tab stops just press **Alt-S**. If you wish to return **Tab** to its normal Framework II operation, press **Alt-Q**. For more information on keyfilters refer to the March 1985 issue of ***TechNotes***, starting on page FW-4.

Printing Program Listings

by Jack Hirsch

Sometimes you need to list to the printer all the command files in a subdirectory or on a particular disk. One option which is readily available to you is the TYPE command. From either PC/MS-DOS or dBASE III you can use the TYPE command to output individual files. Multiple file output, however, is not possible without writing a DOS batch file or a dBASE III program, both of which would require programming time.

A second option is to run the program listed below, Prgprint.PRG, listed below which prints command file listings in alphabetical order.

To set up Prgprint.PRG, first CREATE a database file called Shape.DBF with the following structure which will serve as a structure for your command file's database file.

Structure for Shape.DBF:

Field	Type	Length	Decimals
File	Character	8	

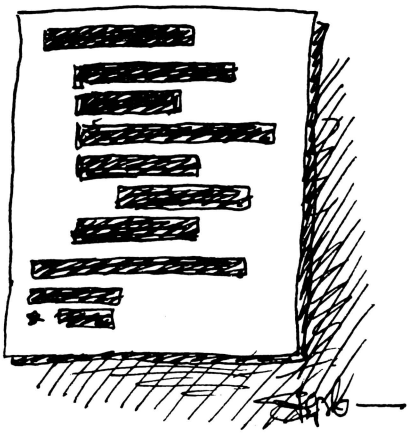
The structure for Shape.DBF is very simple. It has one character field with a length of eight. After you CREATE it, be sure not to add any data to it.

Next let's look at the program itself. It uses two command files. The first one is the command file Prgprint.PRG which is just a menu front-end for its procedure file Fileproc.PRG.

```
* Program.: Prgprint.PRG
* Author..: Jack Hirsch
* Date....: March 1, 1986
* Note....: Setups and menu selections for Fileproc.PRG.
*
SET TALK OFF
SET BELL OFF
SET SCOREBOARD OFF
SET PROCEDURE TO Fileproc
* ---The next three SET commands are optional.
SET INTENSITY OFF
SET DELIMITER TO "[]"
SET DELIMITER ON
* ---Main program loop.
DO WHILE .T.
   CLEAR
   menuchoice = 0
   @ 10,32 SAY "P R G P R I N T"
   @ 14,23 SAY "0. Quit"
   @ 16,23 SAY "1. Create Command Files List"
   @ 18,23 SAY "2. Generate Command File Listings"
   @ 23,35 SAY "Select " GET menuchoice PICTURE '@B' RANGE 0,2
   READ
```

```
   choice = STR(menuchoice,1)
   DO CASE
      CASE choice = "0"
         CLEAR
         SET TALK ON
         SET SAFETY ON
         SET INTENSITY ON
         SET BELL ON
         SET DELIMITER OFF
         CLOSE PROCEDURE
         USE
         RETURN
      CASE choice = "1"
         DO Makefile
      CASE choice = "2"
         DO Fileprt
   ENDCASE
ENDDO * EOP Prgprint.PRG
```

Part two of the program is a procedure file called Fileproc.PRG. In it, the procedure Makefile creates the database file that will serve as the source for the subsequent output of command file listings.



```
* Program.: Fileproc.PRG
* Author..: Jack Hirsch
* Date....: March 1, 1986
* Notes...: Procedure file for Prgprint.PRG.
*
PROCEDURE Makefile
CLEAR
sysname = SPACE(8)
@ 10,10 SAY "Enter the name of the SYSTEM " ;
        GET sysname PICTURE "@!"
READ
* ---If no name is input, return to the menu.
IF sysname = SPACE(8)
   RETURN
ENDIF
sysname = TRIM(sysname)
CLEAR
```

```
* ---Test for existence of the sysname database file.
IF FILE("&sysname..DBF")
   write = "Y"
   SET BELL ON
   ?? CHR(7)
   SET BELL OFF
   @ 23,18 SAY sysname + " already exists - overwrite it? [Y/N]";
    GET write PICTURE "Y"
   READ
   * ---Note that any response other than a Y
   * ---will return you to the menu.
   IF UPPER(write) <> "Y"
      RETURN
   ENDIF
SET SAFETY OFF
ENDIF
@ 23,0
@ 23,19 SAY "Searching for command files - please wait"
RUN DIR *.PRG > Fileprt.TXT
* ---Open the "skeleton" database.
USE Shape
* ---Create a new database file from Shape.DBF.
* ---It will be identical in structure to Shape.DBF
* ---but with the new name that you input as sysname.
COPY STRUCTURE TO &sysname
USE &sysname
* ---Append from the text file.
APPEND FROM Fileprt SDF
* ---Mark for deletion the files that comprise this program as
* ---well as any records containing blank fields. Then PACK the
* ---database file.
DELETE FOR SUBSTR(FILE,1,1) = " " .OR. ;
           SUBSTR(FILE,1,8) = "PRGPRINT" .OR. ;
           SUBSTR(FILE,1,7) = "FILEPROC"
PACK
GO TOP
* ---Check for existence of (.PRG) files.
IF BOF() .AND. EOF()
   CLEAR
   ?? CHR(7)
   @ 21,21 SAY "No command files found - press any key"
   WAIT " " TO null
   CLEAR
   USE
   ERASE &sysname..DBF
ELSE
   * ---Put in alphabetical order by command file.
   INDEX ON File TO &sysname
   * ---Close the database file.
   USE
ENDIF
* EOP Makefile

PROCEDURE Fileprt
* ---Outputs the command file to the screen or the printer.
is_go = .F.
is_prt = .F.
mfile = SPACE(8)
DO WHILE .NOT. is_go
   CLEAR
   @ 5,1 SAY "Enter the name of the file system ==> " ;
         GET mfile PICTURE '@!'
   READ
```

```
   IF mfile = SPACE(8)
      RETURN
   ENDIF
   mfile = TRIM(mfile) + ".DBF"
   * ---Check for existence of the file system
   IF .NOT. FILE("&mfile")
      ?? CHR(7)
      WAIT "File system not found - press any key to retry"
      mfile = SPACE(8)
   ELSE
   is_go = .T.
   ENDIF
ENDDO
@ 8,1 SAY "Output to the printer? [Y/N] ==> " GET is_prt
READ
IF is_prt
   SET PRINT ON
ENDIF
mfile = SUBSTR(mfile,1,AT(".",mfile) - 1)
USE &mfile INDEX &mfile
DO WHILE .NOT. EOF()
   pfile = TRIM(SUBSTR(File,1,8)) + ".PRG"
   TYPE &pfile
   IF is_prt
      EJECT
   ENDIF
   SKIP
ENDDO
IF is_prt
   SET PRINT OFF
   EJECT
ENDIF
* EOP Fileprt
```

As simple and straightforward as it is, Prgprint.PRG can be a useful addition to your library of utility programs.

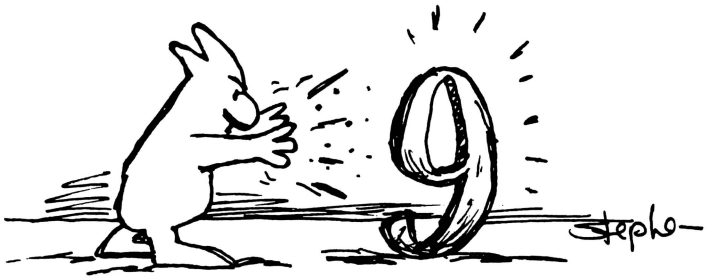
Exchanging Numbers and Text in a Cell

by Gail Kulikowsky

How many times have you found yourself entering a phone number or an address into a spreadsheet or database without first pressing the **Spacebar**? You end up with a strange number or an error constant in the cell. After three or four times your blood pressure begins to rise and you begin to look back at the good old days before computers.

With this in mind, we are going to create a couple of macros to handle the situation. The first macro, **Alt-T**, will take the numeric value of the cell under the cursor and convert it to text. The second macro, **Alt-N**, will take a number that was inadvertently entered as text and convert it to its numeric equivalent.

To set up these macros, go into the Library.Macros frame and create two empty word frames. Label them {Alt-T} and {Alt-N}, including the curly brackets. Place the cursor on the frame border of {Alt-T}. Press **EDIT FORMULA (F2)** and **ZOOM (F9)**, and enter the following program:



```
; {Alt-T} Library macro
; by Gail Kulikowsky
; Converts numeric value into text.

@echo(#OFF),
@performkeys("{F2}{Home}{F6}{End}{Return}{Alt-F7}" &
     "{Esc}{Del} {Alt-F8}{End}{Backspace}{Return}"),
; Note:          ^----- This character must be a space!
@echo(#ON)
```

Now highlight the {Alt-N} frame. Again, press **EDIT FORMULA (F2)** and **ZOOM (F9)**, and enter this macro:

```
; {Alt-N} Library macro
; by Gail Kulikowsky
; Converts a text number into a numeric value.
```

```
@echo(#OFF),
@performkeys(" {Home}{F6}{End}{Return}{Alt-F7}{Del}{Return}" &
; Note:       ^----- This character must be a space!
     "{F2}{Alt-F8}{End}{Backspace}{Return}"),
@echo(#ON)
```

Now when you find yourself with a number where you wanted text, or text where you wanted a number, instead of getting frustrated, just press **Alt-T** or **Alt-N**!

Using Rundbase on a Hard Disk

by Clay Carley

Framework II has given us the capability to run dBASE III from an Apps pull-down menu option. With this new feature, we no longer have to "exit" Framework through a DOS Access frame. When the Apps menu add-in for dBASE III is selected, Framework II loads a frame named Rundbase to the desktop and recalculates it. The formula of Rundbase will check to see if there is enough memory to run dBASE III, set the drive to A:, create an output frame to be used by the **@run** command, and run dBASE III. When you quit dBASE you are returned to Framework II with your desktop documents still intact, and the Rundbase and output frames are automatically deleted.

The above procedure is great for those running floppy systems, but will not work for anyone who has dBASE III installed on a hard disk. If you replace the formula of Rundbase with the one listed here, you can select and run dBASE III from the APPS pull down menu even though it resides in its own subdirectory on the hard disk. Another modification I have made to the original Rundbase program is to eliminate the output frame. Instead, I create a local variable to use with the **@run** command, which eliminates the need to create and delete a frame.

```
; Rundbase
; by Clay Carley

@performkeys("{ctrl-d}c"),
; The following memory check will work only on a system that
; is not using extended memory.
@if( @memavail < 256000,
   @list(
      @eraseprompt,
      @prompt("Not enough memory to run dBASE III"&
           " -- Press a key to continue"),
      @beep,@nextkey,@eraseprompt,@return(#false)
   )
),

; The following statements will select the hard drive as C:, then select
; the subdirectory \DBASE that was created from the root directory,
; and then run dBASE III. Upon QUITting dBASE, the
; default drive and directory are set back to where they were.
@local(output,olddrv,olddir),

; Change this function if dBASE is not on C: drive.
olddrv := @setdrive("C:"),

; Change this function if dBASE is not in \DBASE directory.
olddir := @setdirectory("C:\DBASE"),

@run(output, "dbase.com"),
; Reset default drive and directory.
```

```
@setdrive(olddrv),
@setdirectory(olddir)
```

Note: If you want to run dBASE III PLUS from Framework II, we recommend that you have at least 640K RAM in your machine.

Generating Menus in PC/MS-DOS

by Stephen Goodman

Many sophisticated hard disk management systems have as their central command module a front end that is easy to learn and use. There are front end programs for the PC but, unfortunately, many of these programs are either too expensive to justify the cost, or too difficult to modify to suit the user's needs. Worse yet, some of these front ends remain resident in memory while other programs are running, taking up as much as 40K of RAM. For users with limited or fully allocated memory, this can be a source of difficulty, particularly if you have memory-resident utilities such as Sidekick or Prokey in memory.

You can, however, construct your own custom menu system, for your hard disk, using DOS batch files. Batch files do not remain RAM resident, take up very little disk space, and are easily modified. In order to construct your menu system, all you need is a text editor to create the batch files and DEBUG.COM.

The menu system that follows consists of three files that perform three basic functions of a menu program. The first file, MENU.BAT, is the controlling program that displays the menu screen, gets the menu choice, and selects the appropriate action depending on the user input. The second file, MENU.DAT, contains the menu screen and is TYPEd to the screen by MENU.BAT. This approach is faster than using DOS's REM or ECHO statements to display the menu choices. The third file, CHOOSE.COM, waits for a key input and returns a DOS ERRORLEVEL back to MENU.BAT depending on the key the user pressed. This makes it possible to use the batch commands IF and GOTO to control what actions are to be taken for each possible menu selection.

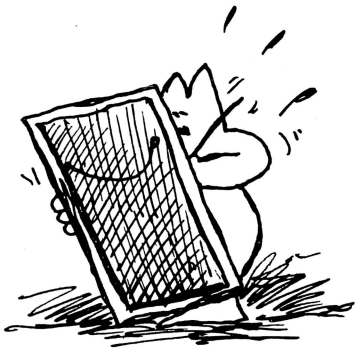
Place MENU.BAT in your AUTOEXEC.BAT file so the menu system executes when the system starts up. MENU.BAT displays user choices, waits for a keypress, and then executes a series of DOS programs. When the commands are completed, control is returned to MENU.BAT, and the cycle begins again.

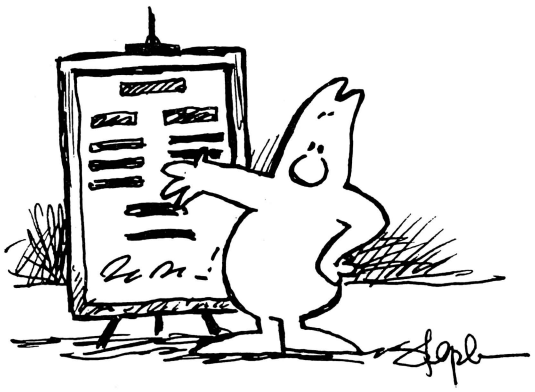
Explanation of MENU.BAT

Following is a summary of the commands used in MENU.BAT. IF and ERRORLEVEL are used together to determine whether an exit code has been returned, and the value of that code. In the same line, place the command to execute if the condition is true. For example, the following command would clear the screen if true:

```
IF ERRORLEVEL 2 CLS
```

Unfortunately all of the commands that need to be executed cannot be put on a single line with this use of the IF command. So, by executing a series of IF ERRORLEVEL statements, we can examine the returned code from CHOOSE.COM for a range of values. If the code is anything other than 1 through 8, the menu is displayed again. Pressing the **Return** key returns the value 9 from CHOOSE.COM and causes you to exit the menu.





This use of the IF command in DOS is similar to the IF statement in dBASE, but requires no ENDIF. However, instead of nesting the execution path as we would in dBASE, we have to utilize the GOTO command to branch to the block batch code to be executed for the menu choice. GOTO works by branching execution of the batch file to the line of code following the specified label. Labels in a batch file always begin with a colon (:), as :CHOICE1. Labels can also be used to create a loop. Instead of having it execute over and over again, a GOTO statement can point to a label, for example :REPEAT, at or near the beginning of the batch file. The result is the batch file equivalent of a DO WHILE .T. loop in dBASE.

When we get to the label chosen, we can have a series of commands following it. The customary method is to have one command after another, one per line, as in:

```
CLS
CD\DBASE
DBASE ACCOUNTS
CD\
```

A problem with this methodology is that this is time-consuming, since each line of the code must be read separately. A more efficient method is to use the FOR command structure. The same series of commands looks like:

```
FOR %%f IN (CLS CD\DBASE DBASE:ACCOUNTS CD\) DO %%f
```

Using this one-line notation is faster than the previous example since only one line of the batch file is acted upon. The parameter %%f is a variable that is sequentially set to each item in the set made up by the commands enclosed by parentheses, which are then executed. Note that each DOS command is equivalent to a command line at the DOS prompt and has no spaces in it. Colons replace the spaces, as in the case of "DBASE ACCOUNTS."

To display the batch menu screen contained within MENU.DAT, we use the TYPE and the CLS command to clear the screen prior to that. But, after the choice is made using keyboard input through CHOOSE.COM, it is good practice to have a message displaying what is going on, particularly if the screen is blank for any length of time. ECHO is used for this. You have probably seen the command ECHO OFF prior to a program's execution. The purpose of this is to hide the echoing back of commands, making a cleaner screen. The command ECHO without any characters after it displays the present status of ECHO. But if it is used after the status is set to OFF, ECHO with a series of characters after it will echo back only the characters. For example, the command "ECHO Loading dBASE III PLUS" returns the message, "Loading dBASE III PLUS."

In addition, if you use ECHO with DOS's redirection ability (using > and <), you can provide your batch file with a way of skipping a line or two down the screen. This is done by ECHOing to a "nonexistent" device called NUL. The proper syntax for this is ECHO > NUL:, but note that, in batch files (as opposed to true device use, such as COPY), the colon can be left out, resulting in ECHO > NUL. Note the use of NUL: in suppressing the output of a program, such as a clock driver, when the program occurs, through appending the > NUL: to the program name. Don't do this with dBASE or other such programs, since all you'll see is a blank screen - but I don't have to go over that. If you have ECHO OFF, on occasion a program can jolt DOS back to ECHO ON status. Therefore, use > NUL: for maintaining OFF status, as in the command below:

```
BRDCLOCK > NUL:
```

You can create batch files that display tasks of a specific job in the menu. This will allow for flexibilty as your job changes over time. Instead of having to go in and change MENU.BAT and MENU.DAT every time a new job is assigned, you can change the file THISWEEK.BAT to suit the job. THISWEEK.BAT could also contain just the one line

```
FOR %%f IN (CLS CD\PROJECT ...etc...) DO %%f
```

to perform the commands necessary to go to the project desired. One problem occurs, however, when executing batch files from batch files. Control will not be returned to a calling batch file upon the called file's completion. The only way to execute a batch file from another is through COMMAND.COM and the /C parameter. The command COMMAND /C THISWEEK.BAT loads a temporary copy of COMMAND.COM into memory, taking up approximately 3K, and in turn executes the batch file THISWEEK.BAT. It might even be preferable to execute another copy of MENU.BAT, perhaps called MENU2.BAT, which would act just like it's companion batch file, but with different choices.

In order to return control to MENU.BAT the last command in THISWEEK.BAT must be EXIT. As with the RUN command in dBASE III, the QUIT TO "COMMAND" in dBASE II, and DOS Access in Framework II, EXIT returns control to the calling program, and relinquishes the memory.

For further reference on the DOS Batch Language, consult your PC/MS-DOS manual under Batch Commands.

Setting Up

First of all, enter your text editor and enter MENU.BAT, as shown below in its entirety, and save to disk. Next, create MENU.DAT. A good text editor to use is SED, the dBASE screen editor, which gives the programmer a good idea of what the screen will look like. If you use SED, be sure to save MENU.DAT as a text file, and not as a series of dBASE @...SAY commands.

MENU.BAT

```
ECHO OFF
CD\
:REPEAT
CLS
TYPE C:\BAT\MENU.DAT
CHOOSE
IF ERRORLEVEL 9 GOTO END
IF ERRORLEVEL 8 GOTO ITEM8
IF ERRORLEVEL 7 GOTO ITEM7
IF ERRORLEVEL 6 GOTO ITEM6
IF ERRORLEVEL 5 GOTO ITEM5
IF ERRORLEVEL 4 GOTO ITEM4
IF ERRORLEVEL 3 GOTO ITEM3
IF ERRORLEVEL 2 GOTO ITEM2
IF NOT ERRORLEVEL 1 GOTO REPEAT
CLS
ECHO Loading dBASE III version 1.1... Please wait.
FOR %%f IN (CD\D3\11 DBASE CD\ GOTO:REPEAT) DO %%f
:ITEM2
CLS
ECHO Loading dBASE III Developer's Release... Please wait.
FOR %%f IN (CD\D3\12 DBASE CD\ GOTO:REPEAT) DO %%f
:ITEM3
CLS
ECHO Loading dBASE III PLUS... Please wait.
ECHO>NUL:
ECHO Place dBASE III PLUS System Disk #1 in Drive A:
PAUSE
FOR %%f IN (CD\D3\20 DBASE CD\ GOTO:REPEAT) DO %%f
:ITEM4
CLS ECHO Loading dBASE II version 2.41... Please wait.
FOR %%f IN (CD\D2\241 DBASE CD\ GOTO:REPEAT) DO %%f
:ITEM5
CLS
ECHO Loading dBASE II version 2.43*... Please wait.
FOR %%f IN (CD\D2\243 DBASE CD\ GOTO:REPEAT) DO %%f
:ITEM6
CLS
ECHO Loading Framework version 1.1... Please wait.
FOR %%f IN (CD\FW FW CD\ GOTO:REPEAT) DO %%f
:ITEM7
CLS
ECHO Loading Framework II... Please wait.
FOR %%f IN (CD\FW2 FW/T CD\ GOTO:REPEAT) DO %%f
:ITEM8
CLS
ECHO Executing This Week's Project...
COMMAND /C THISWEEK.BAT
GOTO REPEAT
:END
CLS
ECHO Enter 'MENU'to return to menu...
```

MENU.DAT

MAIN SYSTEM MENU

DATABASE PRODUCTS	INTEGRATED PRODUCTS
1 - dBASE III version 1.1	6 - Framework version 1.1
2 - dBASE III Developers Release	7 - Framework II
3 - dBASE III PLUS	
	OTHER ACTIVITIES
4 - dBASE II version 2.41	
	8 - This Week's Project
5 - dBASE II version 2.43*	

<Return> - Exit to DOS

CHOOSE.COM

To create CHOOSE.COM, create a text file named CHOOSE.IMP. The text file should contain the following lines exactly as you see them. The comments appear after the semicolons, and are not to be typed in.

```
NCHOOSE.COM
A100
MOV DX,112              ; Point to the message.
MOV AH,09               ; Display option for INT 21.
INT 21                  ; DOS Function Call.
MOV AH,01               ; Keyboard input option for INT 21.
INT 21                  ; DOS Function Call.
SUB AL,30               ; Subtract 30 from the resulting key value.
MOV AH,4C               ; Terminate w/Exit Code option for INT 21.
INT 21                  ; DOS Function Call.
RET                     ; Return to DOS.
E112 '                            Enter Your Selection ',10,'$'
RCX
50
W
Q
```

Remember to place a blank line terminated with a carriage return after the INT 21 instruction. When you have verified and saved the CHOOSE.INP, enter the following command at the DOS prompt to create CHOOSE.COM.

DEBUG < CHOOSE.INP

By directing input from the file CHOOSE.INP, DEBUG will create CHOOSE.COM automatically. To test CHOOSE.COM, simply type CHOOSE at the DOS prompt. If everything you have done is correct, you will see a prompt 'Enter Your Selection' followed by a forward arrow, waiting for your key input. Press any key to return to the DOS prompt.

For our purposes, add 25 spaces prior to the prompt message and its arrow in order to center it on the page. If you choose to change the length of this displayed string, remember to change your input for CX to an appropriate size to accommodate more or less space in the file. Remember, this number as all others in DEBUG are in hexidecimal. Refer to the Glossary in your dBASE III or BASIC manual for help with hexidecimal numbers.

Don't stop reading if you don't think you know anything about assembly language. The application of assembly language here is very direct, simple, and easy to implement. DEBUG.COM is used to construct CHOOSE.COM, the program that returns the ERRORLEVEL to MENU.BAT.

CHOOSE.COM works in three ways:

1. Displays a prompt.
2. Waits for a keypress.
3. Returns an exit code upon termination.

Interrupt 21 (INT 21) is used for all three of these functions. In displaying to the screen, first we point to the address in the program where the string to display will reside. This is done by placing the value of the location, 112, in register DX. The string is entered later with the E command. It must be terminated by a hex 24, or '$'. If the hex value 09 is placed in the AH register and then INT 21 is called, the string will be printed to the screen at the current location on the screen.

Next, in order to provide for user input, we call again upon INT 21. But this time, we place the hex value 01 in register AH. This calls DOS to wait until any key has been pressed, at which time the scan code of the key is placed in register AL. Then the program will continue on to the next instruction.

In order to return an exit code based upon the scan code of the key that can be handled by the batch file—that is, between 1 and 9—we subtract 30 hex from the value returned in register AL. This is done with the SUB AL,30 command. Note that, should the user of the menu press any other key than the numbers 1 through 9, CHOOSE.COM will react differently, since it is passing a higher scan code than the numeric keys do. Thus, the ERRORLEVEL will be different. Take note that the ERRORLEVEL parameter in this case can pass values with more than just 0 through 9; for example, pressing an 'S' would return an ERRORLEVEL value of 35. However, remember that, the more choices you have, the longer the batch file will be, resulting in a much slower run time. See the IBM Technical Reference or BASIC manuals for listings of scan codes.

Next, in order to provide a pipe for the resultant code, still in register AL, place the hex value 4C in register AH. When INT 21 is called again, the value in register AL is passed as an exit code, a kind of result code. We then use the RET to quit to DOS and the program is finished.

Implementation of MENU.BAT

The way I have written MENU.BAT essentially reflects how I use my hard disk at work. As technicians of Software Support, we have to have several versions of our software programs ready to use for debugging purposes. I had been using several batch files to perform the varied functions you see in MENU.BAT. Instead of having to keep watch over several batch files, I decided to consolidate them into one batch file.

For this example, I have placed MENU.BAT, MENU.DAT, and CHOOSE.COM all in a subdirectory called \BAT. The PATH in DOS is set to that directory, among others.

Therefore, to use this series of programs together, enter MENU at any DOS prompt, or include MENU as the last line in your AUTOEXEC.BAT file. Every time a task is completed, as well as when the menu is terminated, the user is returned to the root directory of the hard disk. This provides a common reference for use of the hard disk, as well as a practical interface for users who would rather not have to enter the same DOS commands manually every day.

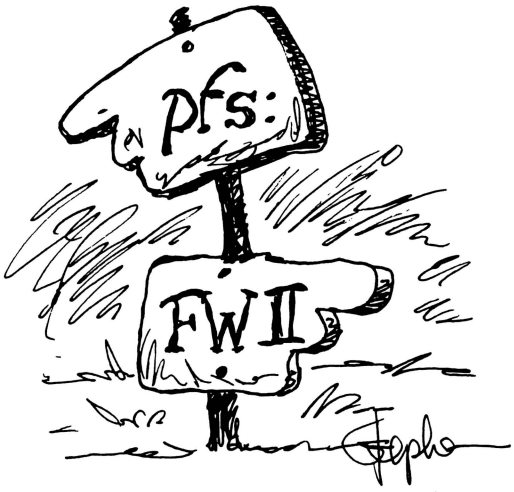
Converting PFS Files to Framework II

by Quinn Wildman

In the August 1985 issue of ***TechNotes***, Steve Silverwood and I wrote an article on how to transfer data from *pfs:file* to dBASE III. In this article, I discuss how to make a similar transfer to Framework II.

The problem in going from *pfs:file* to Framework II is that *pfs:file* does not generate a file that Framework II can directly read. So, you must generate a *pfs* print file just as we did before for dBASE III.

The *pfs* print file contains at least one line per field with variable field lengths. To be read by Framework II as an ASCII text file and be put into a Framework II database, the print file must have only one line per record. Another problem is that the *pfs* print file places all blank fields at the end of the record. Thus, Framework II has no way of knowing which fields should be blank. Something must be entered in every field so that *pfs:file* positions them correctly in the record.



The procedure for transferring a file from *pfs:file* to Framework II is as follows:

In *pfs:file*:

1. Make sure each field is no longer than one line.

2. All blank fields must contain a "dummy" value. This value should be clearly inappropriate in the context, such as the "x" in the following *pfs* record:

   First Name: Quinn
   Last Name: Wildman
   Address: 20101 Hamilton Ave.
   City: Torrance State: CA Zip: 90502-1319
   Home Phone: x
   Work Phone: (213) 329-8000
   Category: x

   We will use the "x" as a flag to identify blank fields to Framework II.

3. Select print from the main menu.
4. Enter "No" to printing item names.
5. Enter a filename, such as Newfile.TXT, for the print destination.
6. Make the number of lines per page equal to the number of fields that you have.
7. Set the number of copies to one.
8. On the print spec menu, enter an x for every field.

This will generate a file with records in the following format:

Quinn
Wildman
20101 Hamilton Ave.
Torrance
CA
90502-1319
x
(213) 329-8000
x

In Framework II:

1. Import the print file that you created in *pfs:file* as an ASCII text file.
2. Create a database frame with the same number of fields as in your *pfs:file* and at least as many records. The database must be named Db, the first field must be called First, and the last field must be called Last. The other fields may or may not have names at this point. Once you have transferred the data into the database, you can change your field names to anything you wish.
3. Create an empty word frame and label it *pfs*. Then press **Dnlevel** and create a subframe called Transfer. Next, enter the following functions into the formula area of each frame.

PFS:

```
; pfs:file Transfer Utility
; by Quinn Wildman

@setmacro({Alt-T},PFS.Transfer),
@prompt("Highlight the imported ASCII text file, "
          "press Dnlevel, then Alt-T")
```

PFS.Transfer:

```
@local(dummy),
dummy:=@inputline("Enter dummy character"),
@performkeys("{Ctrl-Home}"),
@while(@textselection<>@chr(13),
    @performkeys("{F6}{End}{Return}"),
    @if( @textselection<>dummy,
    @put( Db.First:Db.Last, @textselection)
    ),
    @next( Db.First:Db.Last),
    @performkeys("{Rightarrow}")
)
```

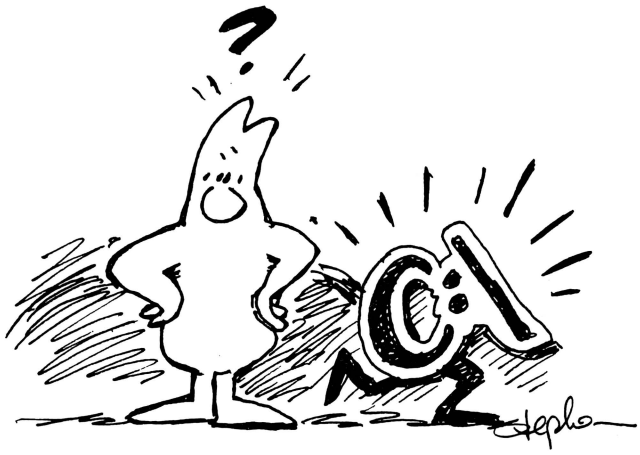
4. Save the *pfs* frame to the disk with **Ctrl-Return**. Press **RECALC (F5)** with the cursor on the *pfs* frame. A prompt appears at the bottom of the screen telling you to select the ASCII text file and press **Alt-T**.
5. Highlight the ASCII text file, press **Dnlevel**, then **Alt-T**.
6. Change the field names in your database, if you wish, and then save it with **Ctrl-Return**.

USAGE TIPS

dBASE II

by Christopher White

Using Subdirectories Under PC/MS-DOS 3.1



dBASE II does not recognize subdirectories under PC/MS-DOS versions 2.0 and above. If you are running dBASE II under PC/MS-DOS version 3.1, however, you can work around this limitation. Version 3.1 has the command SUBST which will assign a drive letter to any subdirectory, creating a new logical drive. Any disk operations including both reading and writing to the new logical drive will actually go to the substituted subdirectory. This will allow dBASE II to access various subdirectories by using the logical drive designator to specify the subdirectory.

To set this up:

1. At the DOS version 3.1 prompt type:

   SUBST D: C:\<your directory>

2. To access the subdirectory in dBASE II, SET DEFAULT TO <new logical drive>.

   Any file operations that refer to the logical drive will now operate in the assigned subdirectory. This includes any commands that create files, give statistics on files, or use files.

Left-Justifying Numerics

To left-justify numeric variables, convert them to character type and trim the leading blanks. The expression

STR(<variable>, 10 - @("-",STR(-1 * <variable>,10,0)),0)

Maximum length of numeric variable

Maximum decimal places of numeric variable

will accomplish this. You can place it in any command that supports the results of a character expression. This includes the REPORT FORM, ?, and @...SAY commands.

To use this expression please note that you must specify the maximum length of the numeric variable in two places. The length includes all the whole number digits, the decimal point, and decimal digits. If you have decimal digits, be sure to specify the number of places as noted above.

dBASE III

by Christopher White

ASSIST/HISTORY

There is a subtle but powerful synergy between ASSIST and the dot prompt in dBASE III PLUS. All commands executed through the ASSIST mode are recorded into HISTORY if SET HISTORY is ON. This feature allows you to create command lines in ASSIST, and then have them available for re-edit and re-execution at some later time.

In the simplest sense, ASSIST can be viewed as a command line builder. If you do not know the syntax of a given command or simply do not want to type it, ASSIST provides what its name implies, prompting. For a series of commands that comprise an action, ASSIST provides prototyping and code generation capability. By LISTing HISTORY TO a (.PRG) file a session in ASSIST is transformed into editable program file. From a broader perspective, ASSIST can be viewed as a kind of applications generator. Each series of steps you go through to perform a logical action defines the skeleton or outline of an application, no matter how simple. By capturing the set of steps into a (.PRG) file via the dot prompt, an action executed in ASSIST can be into expanded into a full-blown application or merged into existing applications.

For instructions on how to LIST the contents of HISTORY into a command file, see page 8 of the February 1986 issue of ***TechNotes***.

Framework II

by Kent Irwin

Subscripts and Superscripts

If your printer can print superscripts and subscripts, then Framework II will probably support the use of these modes in your documents. The following chart shows the ASCII codes that are used by Framework II for this feature.

ASCII Decimal Code	Mode
244	Superscript ON
245	Superscript OFF
246	Subscript ON
247	Subscript OFF

To use one of these modes in a document, begin by putting the cursor on the first character you want scripted. Then hold down the **Alt** key and type the appropriate code from the chart above. Next, release the **Alt** key and a special character will appear in the frame. Now use the same process to insert a code to shut off the scripted mode at the appropriate place in your document.

Always use the codes from the chart above, regardless of what printer you have. If these codes do not seem to work for you, then your Framework printer driver probably does not support this feature.

Importing Framework 1.x Files

Framework II is unable to make use of virtual memory when importing Framework 1.x files. If there is not enough real memory to import a large 1.x file, you must either delete some work on the desktop, or segment the file (using your old version of Framework). You can then load the smaller files and put them back together in Framework II.

Another way to gain additional memory is to delete the contents of the Library. This will save you a certain amount of memory, depending on how large your Library is. To do this, press **Alt-F1**, **EXTEND SELECT (F6)**, **End**, **Del**. Be very careful, however, not to save the empty library. When you select **Quit** from the Disk pull-down menu, Framework II will ask you "Library has been changed, Save (y/n)?" You must answer "N" to this question to preserve the library currently on your disk.

Memory Requirements for Extended Memory

As the documentation states, Framework II requires 384K of memory to load. To use the extended memory feature, however, you must have 512K so that Framework II is able to initialize the extended memory driver. In addition, if you have 512K of memory you cannot have Extended Memory ON and Keep Overlay ON at the same time. On a 640K system both of these options may be selected.

If you are getting the following error message on a 512K sytem:

> "Not enough real memory to load extended memory driver."

then some other program is using a part of your RAM. Check for resident programs (SideKick, ProKey, and other utilities) in your DOS AUTOEXEC.BAT file. These programs will have to be removed from memory before you can use Framework II with extended memory.

SIZE (F4) with Multiple Columns

To change the size of several adjacent spreadsheet or database columns at one time, highlight all the columns with **EXTEND SELECT (F6)** and press **SIZE (F4)** to begin the sizing operation. When you have adjusted the columns to the width you want, you can press **Return** to accept the new size, or you can press **Esc** to abort. At this point, pressing **Esc** will return only the first column to its original size. All of the other selected columns will retain the new size.

Copying Text Containing Fractions into Spreadsheets

Copying a value such as "15 1/8" from a text frame to a spreadsheet cell will result in a value for that cell of 15.125. However, copying a value like "15 1/16" (or any fraction with a two or more digit denominator) into a cell will cause that cell to have a value of 15, and the cell to the right to contain the text "1/16".

Framework II handles fractions like this because the text-to-spreadsheet copy was designed to handle data downloaded from stock services (like Dow Jones News/Retrieval Service), which use fractions no smaller than eighths. For more information about this "Dow Jones rule" see page 4-28 of *Using Framework II.*

Removing the Last Column Referenced by @sum

Removing the last column/row of cells referenced by an **@sum** formula will return #N/A to the formula cell. This is because Framework II cannot calculate a sum with no ending point, and the formula is not automatically modified to recognize the removal.

The formula will be adjusted and will return the proper value as long as the last column/row referenced by the **@sum** is not the one removed.

It should be noted that this is typical spreadsheet behavior—several other popular spreadsheets were tested and performed identically.

Personal Dictionaries

The first time you save a personal dictionary to disk, it may take a long time and generate a very large file. This is because the dictionary's internal format is not optimized for use by the spell checker until the first time the dictionary is read from the disk.

The solution is to save the file using the **Write Personal Dictionary File** option on the **Spell Check** submenu, then load it back onto the desktop and save it a second time without changing anything in the frame. This will cause the dictionary to be optimized, resulting in a much smaller file.

A personal dictionary file can be loaded to the desktop in the same way that an ASCII text file can. Simply put the cursor on the name of the dictionary file in the drive cabinet, and press **Return**. You can edit this dictionary frame (which is actually a word frame when it's on the desktop), as long as you only put one word per line. To save the dictionary, make sure the cursor is on the dictionary frame and select **Write Personal Dictionary File** from the **Spell Check** submenu. Your dictionary will be saved with the frame label as its filename, and .DCT as its extension.

Mailmerge/Labels

There are several options on the **Print Output Options** submenu which have no effect on **Mailmerge** or **Print Labels**, and may in fact appear not to function. For instance:

1. **Start on Page**: If this option is set to a value greater than 1, then no mailing labels will be printed. This option should only be used with Mailmerge when you have a multi-page letter and not all of the pages are to be printed.
2. **End after Page**: No effect on labels, and should only be used with Mailmerge when not all pages of the the letter are to be printed.
3. **Number of Copies**: This also has no effect; it will not cause Framework II to print multiple copies of a form letter or mailing label.

Renaming .PAR Files to .TCM Files

The proper way to convert .PAR files to .TCM files is to use the **Load Setup File** option in the **Apps:Telecommunications:Setup** submenu to get the .PAR file loaded into memory. Then use the **Write Setup File** option from the same menu to save it as a .TCM file.

Simply renaming the .PAR file to a .TCM file will not work correctly, as there are internal differences between the two types of files.

LETTERS

Using Hard Tabs in SED—The dBASE Screen Editor

Dear ***TechNotes***:

In "SED—The dBASE Screen Editor," page 22, the December 1985 issue of ***TechNotes***, Karen Robinson mentions the hard tab inserted by SED. She suggests a work-around of using spaces instead of **Tab**s in SED. Unfortunately, SED creates a hard tab for every 8 spaces even if you don't use the **Tab** key.

The hard tab is a real pest. Material within TEXT...ENDTEXT is displayed incorrectly if created in SED and more than 8 spaces begin any line. Is there a way to re-convert those little o's to spaces or normal tabs?

J.A.

Dear J.A.:

SED automatically puts in a tab character, ASCII code 9, for each 8 spaces indented. dBASE traps the tab character and interprets it as a space, ASCII code 32. The only work-around for TEXT...ENDTEXT is to use the ? command instead. For example, if your program read as follows:

```
TEXT

        MAIN MENU

        0 - Exit
        1 - Reports

ENDTEXT
```

This would be translated to:

```
?
? "       MAIN MENU"
?
? "       0 - Exit"
? "       1 - Reports"
?
```

Removing Preceding Blanks from Character Strings (dBASE III)

Dear *TechNotes*:

I am using dBASE III version 1.1. Whenever I use the TRIM() function, it removes the trailing blanks from character strings. Is there any way to remove preceding blanks from character strings?

R.G.

Dear R.G.:

The Developer's Release and dBASE III PLUS have a function, LTRIM(), which removes preceding blanks from character strings. Although this function does not exist in dBASE III versions 1.0 and 1.1, the same end can be reached through program control. Assuming "mvar" is a character string which may or may not contain preceding spaces:

```
DO WHILE SUBSTR(mvar,1,1) = " "
    mvar = SUBSTR(mvar,2)
ENDDO
```

The loop is entered only if the first character is blank. If the first character is blank, then the first space is dropped and the remainder of the string is STOREd back into itself. The loop will terminate when the first non-blank character is reached.

Using Two Database Files with LABEL FORM

Dear *TechNotes*:

We are using dBASE III for our mailing lists. To save time while inputting, we enter only the person's name, address, and zip code. From the zip code, we reference a second database file which contains the zip code, city, and state. Is it possible to print mailing labels with the LABEL FORM command using two different database files?

R.H.

Dear R.H.:

Using the LABEL FORM command in combination with the SET RELATION TO command allows you to print labels using information from two or more database files. The SET RELATION TO command allows you to link two database files based on a common key field. In our example, this common, or key, field is Zipcode. The Cities database file contains the fields City, State, and Zipcode, must be INDEXed on the key field, Zipcode. Before CREATEing the LABEL FORM, the database files must be opened and the RELATION must be SET. Assuming our two database files are called Names.DBF and Cities.DBF:

```
SELECT 1
USE Names
SELECT 2
USE Cities
INDEX ON Zipcode TO Zip
SELECT 1
SET RELATION TO Zipcode INTO Cities
```

Now that the environment is set, type:

```
CREATE LABEL Test
```

Use the first screen of the LABEL FORM to define the type of label and the second screen to define the label contents, as follows:

```
Name
Address
B->City + ", " + B->State + "  " + Zip
```

Since fields from the second work area are used, they are referenced by either Cities-> or B->. These are ALIAS names that are assigned to the file in the second work area. The letter "B" is used as it is the second letter in the alphabet. If Cities.DBF were opened in work area 9, "I" would be one of its ALIAS names, the ninth letter in the alphabet. (For further information on using an ALIAS, refer to your manual, or see page 85 of the December 1985 issue of ***TechNotes***, "Top 10 dBASE III Support Questions.")

Writing Large Assembly Language Programs with Framework II

Dear ***TechNotes***:

I am using the word processor in Framework II to write my assembly language programs. I save these frames to disk as DOS text files. I know that the maximum size of a text frame is 65,535 bytes (64K). However, one of my programs is going to exceed 64K. Is it possible to have a frame larger than 64K? I really do not want to split up my program into two files when I write it into a DOS text file.

J.R.

Dear J.R.:

It is not possible to have a frame larger than 64K in Framework II. However, a containing frame may hold more than one 64K frame. When one frame becomes full (or near full), move to the second frame and continue. When you have finished, press **Uplevel**, highlight the border of the containing frame, and write the file out as a DOS text file. This will produce one large text file containing your entire program.

(Direct your technical correspondence to Ashton-Tate Software Support, 20101 Hamilton Avenue, Torrance, CA 90502-1319.)

TRAINING CENTERS

To insure that you have the training you need when you need it, Ashton-Tate has established Authorized Training Centers. These third-party trainers offer courses on all Ashton-Tate/MultiMate software. The courses range in scope from introductory to advanced programming levels. The courseware they use are either Ashton-Tate training materials or materials screened and approved by Ashton-Tate.

Many of the Training Centers provide specialized training, tailored around your individual educational needs, as well as on-site training. On-site training is an excellent method of learning how to use Ashton-Tate products in real-world corporate and small business environments. Consulting services are also available.

If you would like more information, or if your company is interested in becoming an Ashton-Tate Authorized Training Center, call our Educational Sales Representative at (213) 329-8000.

If you or your end users would like to be trained by one of our existing Authorized Training Centers, a listing of their locations follows. New training centers are continually being added. So please call us for the latest information if you do not find a convenient location below.

This list of the Authorized Training Centers has been updated from the January 1986 issue of ***TechNotes***. A new, updated list will appear in future issues of ***TechNotes*** as more Authorized Training Centers are established.

ARIZONA

Drum Research Associates, Inc.
Tucson, AZ
(602) 296-7900

CALIFORNIA

Computer Experience
Costa Mesa, CA 92626
(714) 432-9000

Computergap, Inc.
Foster City, CA 94404
(415) 345-4801

Computer Solutions
Los Angeles, CA
(213) 277-5311

Tek/Link
Mountain View, CA
(415) 948-4300

Mark Computer Technologies
San Francisco, CA 94123
(415) 776-4200

The Training Company
San Francisco, CA 94133
(415) 986-1121

OnLine Computers Plus
Santa Monica, CA
(213) 458-1652

Computer Experience
West Los Angeles, CA 90064
(213) 477-4094

CANADA

Lemmex & Associates, Ltd.
Ottawa, Ontario
(613) 727-7074

CONNECTICUT

Entré Computer Center
Groton, CT
(203) 445-5445

Business Communication Consultants
Stamford, CT
(203) 327-1285

CPCE
Trumbull, CT
(203) 268-0049

Olsten Office Automation School
Trumbull, CT
(203) 268-9300

MicroAge of Rocky Hill
Rocky Hill, CT
(203) 529-5161

COLORADO

Computer Learning Source
Colorado Springs, CO 80907
(303) 590-1573

DISTRICT OF COLUMBIA

Executive Computer Training Center
Washington, DC (area) 22180
(703) 821-3932

Future Enterprises, Inc.
Washington, DC
(202) 662-7676

FLORIDA

University of Miami Continuing Education
Coral Gables, FL 33124
(305) 284-5419

GEORGIA

ExecuTrain
Atlanta, GA
(404) 953-4000

MicroTrain
Atlanta, GA 30328
(404) 255-4420

Software Atlanta
Atlanta, GA
(404) 934-0924

ILLINOIS

Computer-Workshop
Downers Grove, IL 60515
(312) 971-0004

Computer-Workshop
Chicago, IL 60604
(214) 971-0004

Micro Computer Learning Center
Chicago, IL
(312) 332-0419

Professional Training Centers
Hinsdale, IL
(312) 920-0750

KANSAS

Computer Learning Source
Overland Park, KS 66212
(913) 649-1636

MARYLAND

OfficePro
Rockville, MD
(301) 468-3312

Wordpro, Inc.
Rockville, MD 20852
(301) 984-3700

MASSACHUSETTS

Computer Tutor Corporation
Boston, MA 02116
(617) 237-1840

Executive PC Education Center
Boston, MA
(617) 723-2055

Mark Computer Technologies
Burlington, MA 01803
(617) 273-5638

Computer Training Solutions
Stoneham, MA
(617) 438-1510

Team Computer Systems
Watertown, MA
(617) 924-1995

Computer Tutor Corporation
Wellesley, MA 02181
(617) 237-5638

MICHIGAN

GMI Engineering & Management Institute
Flint, MI 48502-2276
(313) 726-9866

Computer Automation Training Center
Lansing, MI 48906
(517) 372-5155

Inacomp Novi
Novi, MI
(313) 348-8841

MISSISSIPPI

Entré Computer Center
Jackson, MS 39201
(601) 353-4488

NEW JERSEY

DB Technology
Cranford, NJ
(201) 276-7474

Executive Computer Network
Fairfield, NJ 07006
(201) 431-5186

Oasis—Instruction Specialists
Freehold, NJ 07728
(201) 431-5186

Computer Applications Learning Center
Morristown, NJ
(201) 539-6050

CES Training Corporation
River Edge, NJ
(201) 599-9510

NEW YORK

Logical Operations
Buffalo, NY
(716) 856-0596

Entech Computer Systems
Commack, NY
(516) 543-3352

Data Tree Associates
New York, NY
(212) 410-0051

IBIS
New York, NY
(212) 794-9271

MicroTrek Enterprises Inc.
New York, NY
(212) 286-0894

Instamation
Rochester, NY 14610
(716) 461-1800

OHIO

Mentor Technologies
Columbus, OH
(614) 262-8041

Micro Center
Columbus, OH 43221
(614) 481-8041

Computrain
Independence, OH
(216) 447-4430

Ernst Business Computer Training
Marysville, OH
(614) 666-3411

PENNSYLVANIA

Word Systems Associates
Mt. Lebanon, PA
(412) 343-4044

PC Concepts
Philadelphia, PA 19103
(215) 557-7895

ICM Professional Services Group
Pittsburgh, PA 15222
(412) 261-3848

Micro Training Specialists Company
Pittsburgh, PA
(412) 687-6311

PC Concepts
Wayne, PA 19087
(215) 687-5554

TEXAS

Complete Computer Systems
Dallas - Ft. Worth, TX 75028
(213) 477-4094

Soft Touch Technology
Plano, TX
(214) 233-5161

Computer Aspects
San Antonio, TX
(512) 657-7981

VIRGINIA

The Trace Group
Harrisonburg, VA 22150
(703) 433-0966

The Trace Group
Richmond, VA 23219
(804) 649-0066

Computer Learning Center/Washington
Springfield, VA 22150
(703) 971-0500

Executive Computer Training Center
Vienna, VA
(703) 821-3932

WISCONSIN

Computer Assistance
Waukesha, WI 53186
(414) 784-7668

UPDATE POLICIES

dBASE II

The latest version of dBASE II, version 2.43*, is now available for update to all registered users of dBASE II, subject to format availability.

In order to update your current version of dBASE II, you must have your original software registration card on file with Ashton-Tate Customer Service. If you are not a registered user, you may request that a registration card be sent to you.

Prices for this update are outlined below, depending on which version of dBASE II you are currently using. (Funds must be in US dollars drawn on a US bank.)

Version 2.3
$75.00 within US - UPS shipping
$90.00 within US - DHL (3 day)
$100.00 in Canada - DHL (only)

Version 2.4
$50.00 within US - UPS shipping
$60.00 within US - DHL (3 day)
$75.00 in Canada - DHL (only)

Version 2.41 (purchased before 12/1/84)
$25.00 within US - UPS shipping
$35.00 within US - DHL (3 day)
$40.00 in Canada - DHL (only)

If your product was purchased after April 1, 1985, the 2.43* update is at no charge. Please include proof of purchase with requests for free updates. Direct all correspondence to:

Ashton-Tate Customer Service
dBASE II Update
20101 Hamilton Ave.
Torrance, CA 90502-1319
(213) 329-1664

dBASE III PLUS

Registered users of dBASE III version 1.0 or 1.1 will be notified by mail regarding the upgrade to dBASE III PLUS. **Unregistered** users of dBASE III version 1.0 or 1.1 who wish to upgrade to dBASE III PLUS are required to send in their registration card and contact: (213) 329-1664, or (213) 329-9989.

Framework II

Registered users of Framework version 1.0 or 1.1 will be notified by mail regarding the upgrade to Framework II. **Unregistered** users of Framework version 1.0 or 1.1 who wish to upgrade to Framework II are required to send in their registration card and contact: 1 (800) 437-4329.

Disk Replacement Policy

In the event that you find a defect in the programming, materials, or workmanship of an Ashton-Tate system disk within three months of purchase, we will replace it free of charge. Call Software Support first to verify the defect, (213) 329-0086, between the hours of 6:00 am and 5:00 pm, PST.

If you damage your system disk anytime within three years of purchase (and after the 90-day warranty), there is no need to call Software Support. Our Customer Service department will replace it for $20.00 (per disk), payable by check, money order, or VISA/MasterCard (U.S. funds drawn on U.S. bank). Just complete the replacement form below and send it to Customer Service at the address below, along with the defective disk and your payment. (Please enclose a dated proof-of-purchase.)

Ashton-Tate Customer Service
Disk Replacement Form
20101 Hamilton Ave.
Torrance, CA 90502-1319

PLEASE BE ADVISED: You **must** be **registered** with Ashton-Tate in order to receive a replacement disk. If you have lost your registration card, or feel that you may not be on file, please call Customer Service at (213) 320-1664, Monday through Friday, 8:00 am to 5:00 pm, PST.

DISK REPLACEMENT FORM

Name _____

Company _____ Dept./Div. _____

Address _____ Suite/Room _____

City _____ State _____ Zip _____

Country _____ Phone ( )_____-_____

* System Disk Serial Number (REQUIRED) _____

Ashton-Tate Product Used _____

Computer Brand and Model Used _____

AMOUNT ENCLOSED: $_____

NOTE: You must either insure any defective system disk being returned or assume the risk of loss or damage in transit.

* If your disk indicates your serial number as being "ON FILE," refer to your original system disk or system backup disk for your serial number.

READER SUBMISSION FORM

If you have a programming tip, a handy formula, or a short utility, and you are interested in sharing it with other dBASE II, dBASE III, and Framework enthusiasts in a future issue of ***TechNotes***, please read the provisions below. After reading these guidelines, fill out this form and send it along with your submission(s) to the following address. Submissions must be made on disk (IBM or compatible format) in standard ASCII format or in Framework (.FW) format.

TechNotes Submissions
Software Support Center
Ashton-Tate
20101 Hamilton Ave.
Torrance, CA 90502-1319

NAME _____ DATE _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

PHONE (During normal business hours) ( ) _____ - _____

(SIGN HERE TO ACKNOWLEDGE YOUR UNDERSTANDING OF THE PROVISIONS BELOW)

To establish a clear understanding of the rights and usage of submissions, you are asked to first represent and warrant that:

1. You are the author of the material: that it is original and has not been heretofore published; that it does not infringe upon any copyright, proprietary right or any other right whatsoever; that it contains no matter that is contrary to law; and that you are the sole owner of the rights herein conveyed to us, and that such rights are free and clear.
2. We shall have the right to use your name, biography or other information pertinent to your submission.
3. We shall have the right, at our discretion, to edit the material to suit the requirements of our publication.
4. Suggestions, comments, and programming materials submitted to ***TechNotes*** are not for reply. Materials submitted for publication are considered contributions to the "pool" of information found in ***TechNotes*** for the advantage of our readers. Although, we do not pay or in any other way compensate you for your contribution, your name shall appear as author or contributor.

Thank you for reading ***TechNotes***. If you do not have a submission for us, write us anyway and let us know what you think about our publication. We certainly appreciate any feedback you may have.

TechNotes Reader Survey

In our effort to make ***TechNotes*** an even more valuable resource to you, we want your thoughts about it. Please take a minute to fill in the following questions, and return it to us in the postage-paid return envelope. Share with us your comments so that we may keep ***TechNotes*** in tune with your needs.

= =

1. Which of the following statements best describes your *overall feeling* about ***TechNotes***?

   Excellent, almost always has useful information for me. ____
   Good, frequently has useful information for me. ____
   Average, occasionally has useful information for me. ____
   Fair, seldom has useful information for me. ____
   Poor, almost never has useful information for me. ____

2. Which Ashton-Tate product(s) do you use one or more times a week? (Check all that apply.) ____None

   ____ dBASE II ____ dBASE III ____ Developer's Release
   ____ dBASE III PLUS ____ Framework ____ Framework II

3. How long have you used Ashton-Tate products? ____ yrs ____ mos.

4. How long have you subscribed to ***TechNotes***? ____ yrs ____ mos.

5. Do you keep your back issues of ***TechNotes***? ____ Yes ____ No

6. How many people, *besides you*, read your copy of ***TechNotes***?____

7. Please rank from 1 to 8 the following types of information in order of their *usefulness to you*. (1 is most useful, 8 is least useful.)

   ____ Technical Articles ____ Anomalies
   ____ Application Articles ____ Sample Programs
   ____ Letters from users ____ Update Policies
   ____ Usage Tips ____ Your Choice ______________________________

8. When you have a technical question or problem related to Ashton-Tate products, what do you do *first*?

   ____ Call Software Support
   ____ Read ***TechNotes***
   ____ Connect to On-Line Services (CompuServe, The Source, etc.)
   ____ Read the documentation
   ____ Call a knowledgeable friend
   ____ Other ____

9. Which MultiMate® product(s) do you use? (Check all that apply.)

   ____ None
   ____ MultiMate Advantage™ ____ On-File™
   ____ MultiMate 3.3 Series ____ GraphLink™
   ____ MultiMate Executive™ ____ Business Advantage™ Keyboard
   ____ Just Write™ ____ Business Advantage Communications Board

10. Would you like to see topics about MultiMate software covered in ***TechNotes***? ___ Yes ___ No

    If yes, what topics?

    1._____

    2._____

11. Would you like to see *advertisements* in ***TechNotes*** on products and services related to Ashton-Tate products? ___ Yes ___ No

    If yes, what products or services would you like to see advertised?

    1._____

    2._____

    If no, why not?_____

    _____

12. Would you like to see *case studies* in ***TechNotes*** on how Ashton-Tate software is being used to solve business problems? ___ Yes ___ No

    If yes, do you have a suggestion for a case study?

    _____

13. Which of the following would you like to see expanded in coverage and which reduced in coverage in ***TechNotes***?

	Expand	**Reduce**	**Stay as is**
dBASE II	___	___	___
dBASE III	___	___	___
Developer's Release	___	___	___
dBASE III PLUS	___	___	___
Framework 1.x	___	___	___
Framework II	___	___	___
Usage Tips	___	___	___
Anomalies	___	___	___
Letters	___	___	___
Policies	___	___	___

14. What, if any, additional topics would you like to have included in ***TechNotes***, in order of your preference?

    1._____

    2._____

    3._____

15. What is the single thing that you *like most* about ***TechNotes***?

    _____

16. What is the single thing that you *like least* about ***TechNotes***?

    _____

17. What other comments would you like to make regarding any topic?

    _____

    _____

    _____

Thank you for your valuable input.

CUMULATIVE INDEX

January 1986 through March 1986

Articles

Subject	Product	Issue	Page	Title and Description
@...SAY	**dBASE III PLUS**	Mar	11	**The IIF() Function** *Using this new function for conditional displaying*
@isabend	**Framework II**	Feb	38	**The Protean Powers of @isabend** *Explanation of this powerful function which tests for error conditions*
@panel1/@panel2	**Framework II**	Jan	14	**Overview of New Framework II Features** *Explanation of these new functions*
ABBREVIATIONS	**Framework II**	Feb	24	**Library Macros and Abbreviations** *How to use, edit, and create abbreviations in the Library*
ASSEMBLY LANGUAGE	**dBASE II**	Feb	13	**Interfacing with the Machine** *How to use the assembly language interface with dBASE II*
DATA	**dBASE III**	Feb	31	**Data Encryption Strategies** *Programs for data security in dBASE III*
	dBASE III PLUS	Feb	11	**Toggling Between Table and Form Views** *Program to toggle between table view and form views in dBASE III Plus*
DOS	**Other**	Mar	29	**Generating Menus in PC/MS-DOS** *How to use batch files to create menus in DOS*
END-OF-FRAME	**Framework II**	Feb	40	**The Protean Powers of @isabend** *Using @isabend to test for boundaries of frames*
ERROR CHECKING	**Framework II**	Feb	39	**The Protean Powers of @isabend** *Using @isabend to check for error conditions*
FINANCIAL MODELS	**Framework II**	Feb	34	**Data Tables in Framework II** *Creating data tables for "what-if" modeling*
HISTORY	**dBASE III PLUS**	Feb	7	**Using History in dBASE III PLUS** *Using HISTORY as an algorithm-building tool*
IIF()	**dBASE III PLUS**	Mar	8	**The IIF() function** *Examination of this new dBASE III PLUS function*
IMPORT	**Framework II**	Feb	28	**Framework II for Framework 1.x users** *Importing Framework 1.x files into Framework II*
		Mar	36	**Converting PFS files to Framework II** *Importing files from pfs:file to Framework II*
INDEX	**dBASE III PLUS**	Mar	10	**The IIF() Function** *Using this function to INDEX on logical fields and conditional keys*
INSTALLATION	**Framework II**	Feb	28	**Framework II for Framework 1.x users** *Framework II installation procedures for Framework 1.x users*
INTERVIEW	**Framework**	Feb	3	**Technotes Interview with Robert Carr** *Part 2, Framework creator discusses possible plans for the future*
		Jan	4	**Interview with Robert Carr** *Part 1, the creater of Framework discusses its development*
LABEL FORM	**dBASE III PLUS**	Mar	9	**The IIF() Function** *Using this new function to format output*
LIBRARY	**Framework II**	Feb	24	**Library Macros and Abbreviations** *Introduction to the Library cabinet*
		Jan	32	**Creating String Functions** *Adding functions and macros to Framework II Library*
LOGICAL FIELDS	**dBASE III PLUS**	Mar	10	**The IIF() Function** *Using IIF() to INDEX on logical data*

Subject	Product	Issue	Page	Title and Description
MACLIB	**Framework II**	Feb	28	**Framework II for Framework 1.x users** *Transporting MacLib into Framework II*
MACROS	**Framework II**	Feb	24	**Library Macros and Abbreviations** *How to use, edit, and create macros in the Library*
MAILING LABELS	**Framework II**	Mar	14	**Automating Mailing Labels** *FRED program to ease the generation of mailing labels*
MENUS	**dBASE III PLUS**	Mar	12	**SET COLOR from a Menu** *A program for defining color from a menu*
	Framework II	Feb	21	**Overview of new Framework II features** *New options in the Locate, Frames, Graph, and Print pull-down menus*
		Jan	14	**Overview of New Framework II Features** *New options in the Disk, Create, and Edit pull-down menus*
	Other	Mar	29	**Generating Menus in PC/MS-DOS** *How to use batch files to create DOS menus*
MODIFY STRUCTURE	**dBASE II**	Jan	10	**An Alernative to MODIFY STRUCTURE** *A dBASE II program to emulate this command*
NEW FEATURES	**dBASE III PLUS**	Feb	7	**Using HISTORY in dBASE III PLUS** *Using this new feature as an algorithm-building tool*
		Mar	4	**Understanding Views** *Discussion of views as they enhance relational operations*
		Mar	8	**The IIF() Function** *Application uses of this new function*
		Feb	21	**Overview of New Framework II features** *New Locate, Frames, Graph, and Print pull-down menu options*
	Framework II	Feb	24	**Library Macros and Abbreviations** *Introduction to the Library Cabinet*
		Feb	28	**Framework II for Framework 1.x Users** *Explanation of installation and configuration for users who upgraded*
		Feb	38	**The Protean Powers of @isabend** *Using this new functions to test for error checking*
		Jan	14	**Overview of New Framework II features** *@panel1, @panel2; New Disk, Create, and Edit pull-down menu options*
		Jan	32	**Creating String Functions** *Adding custom functions and macros to the Library*
		Mar	27	**Using Rundbase on a Hard Disk** *Program to run a hard-disk based dBASE III from Framework II*
PFS:FILE	**Framework II**	Mar	36	**Converting PFS Files to Framework II** *Program to transfer the files*
PRINTING	**dBASE III**	Mar	21	**Printing Program Listings** *Program to print dBASE III program files*
RANDOM NUMBERS	**dBASE**	Jan	17	**Random Numbers Generation for dBASE** *Random number generation in dBASE II and dBASE III*
REPORT FORM	**dBASE III PLUS**	Mar	8	**The IIF() Function** *Using this new function to format REPORT FORM output*
ROBERT CARR	**Framework**	Feb	3	**Technotes Interview with Robert Carr** *Interview with the creator of Framework, part 2*
		Jan	4	**Interview with Robert Carr** *Interview with the creator of Framework, part 1*
RUNDBASE	**Framework II**	Mar	27	**Using Rundbase on a Hard Disk** *Program to run a hard-disk based dBASE III from Framework II*
SET COLOR	**dBASE III PLUS**	Mar	12	**SET COLOR from a Menu** *A program which generates a menu for color definition*

Product	Entry	Issue	Page	Description
SETUP	**Framework II**	Feb	28	**Framework II for Framework 1.x users** *Replaces installation and configuration programs in Framework*
SPREADSHEET	**Framework II**	Feb	34	**Data Tables in Framework II** *Creating data tables for "what-if" modeling*
		Mar	25	**Exchanging Numbers and Text in a Cell** *Useful macros for Framework II spreadsheets*
	Framework	Jan	7	**Interview with Robert Carr** *Evolvement of spreadsheets in Framework*
TABS	**Framework II**	Jan	34	**Removing Tabs from a Word Frame** *Program used on text files before or after transmitting via telecom.*
		Mar	18	**Variable Tab Stops** *Creating variable tab stops in Framework II*
TELECOMMUNICATIONS	**Framework II**	Jan	34	**Converting PFS files to Framework II** *Program used on text files before or after transmitting*
		Jan	34	**Removing Tabs from a Word Frame** *Program used on text files before of after transmitting*
VIEWS	**dBASE III PLUS**	Mar	4	**Understanding Views** *Discussion of views as they enhance relational operations*

Usage Tips

Product	Entry	Issue	Page	Description
dBASE II	@...SAY	Jan	37	*Patches to access line 24 with @...SAY in versions 2.4 or higher*
		Jan	37	*"@ row,column SAY ERASE" only clears one line in CP/M-80 formats*
	DEBUG	Jan	37	*Patches to access line 24 with @...SAY in versions 2.4 and higher*
	NUMERIC VARIABLES	Mar	40	*How to left-justify numeric variables*
	SUBDIRECTORIES	Mar	39	*Using logical drive designators as subdirectories with DOS 3.1*
dBASE III	ASSIST	Mar	40	*Commands executed in ASSIST can be recorded in HISTORY*
	COPY	Jan	38	*Method to COPY fields from one database file to another*
	DELETE	Jan	39	*Deleting records based on values in a related database file*
	SET PATH TO	Jan	40	*If SAFETY is ON, some commands will search the path before writing*
		Jan	40	*Commands which use the path to locate a file*
	SET SAFETY ON/OFF	Jan	40	*If ON, some commands will search the path before writing to disk*
dBASE III PLUS	HISTORY	Mar	40	*Commands executed in ASSIST can be recorded in HISTORY*
Developer's Release	DBRUN.COM	Jan	40	*This file, not DBRUN.EXE, should be renamed as per the documentation*
	DBRUN.EXE	Jan	40	*This hidden file will not show up with DIR and cannot be renamed*
	READ	Jan	40	*In Developer's Release, a READ must have active GETs*
		Jan	40	*Cannot issue a READ from within HISTORY in Developer's Release*
	REINDEX	Jan	40	*Must reissue the SET ORDER TO command if you REINDEX*
	SET ORDER TO	Jan	40	*Use REINDEX to SET ORDER TO its original state*

Subject	Product	Issue	Page	Title and Description
Framework II	.PAR FILES	Mar	43	*Converting .PAR files to .TCM files*
	.TCM FILES	Mar	43	*Converting .PAR files to .TCM files*
	@sum	Mar	42	*Cannot remove the last column/row referenced by @sum*
	@time	Jan	42	*Displaying times with the Numbers pull-down menu*
	APPS MENU	Mar	43	*Some Print Output Options have no effect on Mailmerge or Print Labels*
	DICTIONARY	Mar	42	*May take a long time to save the personal dictionary the first time*
	DOW JONES	Mar	42	*Copying fractions from text into spreadsheets*
	EXTENDED MEMORY	Mar	42	*Memory requirements for extended memory*
	FRACTIONS	Mar	42	*Copying fractions from text into spreadsheets*
	IBM 3270t	Jan	44	*This screen driver is found under "All Other IBM Drivers" in SETUP*
	IMPORT	Mar	41	*Importing Framework 1.x files to Framework II*
	LABELS	Mar	43	*Some Print Output Options have no effect on Mailmerge or Print Labels*
	MAILMERGE	Mar	43	*Some Print Output Options have no effect on Mailmerge or Print Labels*
	PAR FILES	Jan	44	*How to read .PAR files from Framework into Framework II*
	PRINTING	Mar	41	*How to print subscripts and superscripts from Framework II*
		Mar	43	*Some Print Output Options have no effect on Mailmerge or Print Labels*
	SETUP	Jan	42	*Selecting screen color parameters in SETUP*
	SIZING COLUMNS	Mar	42	*Changing several spreadsheet or database columns at one time*
	SPREADSHEETS	Mar	42	*Copying fractions from text into spreadsheets*
	SUBSCRIPTS	Mar	41	*How to print subscripts and superscripts from Framework II*
	SUPERSCRIPTS	Mar	41	*How to print subscripts and superscripts from Framework II*
Framework II	TELECOMMUNI-CATIONS	Mar	43	*Converting .PAR files to .TCM files*
	TEXT	Mar	42	*Copying fractions from text into spreadsheets*
	VIRTUAL MEMORY	Mar	41	*Cannot use virtual memory to import Framework 1.x files*

Anomalies

Product	Entry	Issue	Page	Description
dBASE II	APPEND FROM	Jan	45	*Cautions regarding APPENDing numeric data in version 2.43**
	COUNT	Feb	43	*In DOS ver. 3.x, COUNT may take longer to execute the second time*
	ERROR MESSAGE	Feb	42	*" 'TO' PHRASE NOT FOUND": Memory variable corruption*
	MEMORY VARIABLES	Feb	42	*Corruption of numeric memory variables in dBASE II, version 2.43**
	RENAME.COM	Feb	43	*This file is not on the RunTime disk, as the manual states*
	RUNTIME	Feb	43	*RENAME.COM is not on the RunTime disk, as the manual states*

Subject	Product	Issue	Page	Title and Description
dBASE III	BROWSE	Jan	45	*Problems using the SEEK in BROWSE option with SET DELETED ON*
	DBRUN.OVL	Jan	46	*This file may not get copied during Developer's Release INSTALL*
	ERROR MESSAGE	Jan	46	*"Record is out of range": trying to GOTO a very large number*
	FIND	Jan	45	*With numeric INDEX, FIND <cstring> will FIND first zero valued record*
	GOTO	Jan	46	*Attempting to GOTO a very large number may lock the computer*
	INDEX	Jan	45	*With numeric INDEX, FIND <cstring> will FIND first zero valued record*
	INSTALL	Jan	46	*Developer's Release: INSTALL may not copy DBRUN.OVL*
	PARAMETERS	Jan	46	*Cannot wrap PARAMETERS line with semi-colon*
	RUNTIME+	Jan	46	*DBRUN.OVL may not get copied during Developer's Release INSTALL*
	SEEK	Jan	45	*Problems using the SEEK option in BROWSE with SET DELETED ON*
	SET DELETED ON	Jan	45	*Problems using the SEEK option in BROWSE with SET DELETED ON*

Letters

Product	Entry	Issue	Page	Description
dBASE II	DATA DISPLAY	Feb	44	*Old data displaying in a file with zero records*
	NUMERIC ACCURACY	Jan	49	*Versions prior to 2.43* have 10 digits of numeric accuracy*
	REPORT FORM	Jan	48	*Incorrect totals in versions prior to 2.43**
	SED	Mar	44	*Work-around for hard tabs created in SED*
dBASE III	CHARACTER STRINGS	Mar	45	*Removing preceding blanks from character strings in dBASE III*
	ERROR MESSAGE	Jan	48	*"NO FIND": Character index key length must be a multiple of 4*
		Jan	48	*"Internal error, illegal opcode ": Char INDEX must be multiple of 4*
	IMPORT/EXPORT	Feb	44	*Transferring data from Lotus 1-2-3 to dBASE III*
	INDEX	Jan	47	*Character INDEX key length must be a multiple of 4*
	INSTALL	Jan	48	*Problems with Tallgrass: set Duplicate Directories OFF*
	LABEL FORM	Mar	45	*Using two database files with LABEL FORM*
	LOTUS	Feb	44	*Transferring data from Lotus 1-2-3 to dBASE III*
	SED	Mar	44	*Work-around for hard tabs created in SED*
	SET	Jan		*Accessing SET parameters from within program control*
	TALLGRASS	Jan	49 48	*Problems INSTALLing: set duplicate directories OFF*
dBASE III PLUS	LTRIM()	Mar	45	*This function removes preceding blanks from character strings*
Framework II	ASSEMBLY LANGUAGE	Mar	46	*Using Framework II word processor to write assembly language programs*
	MEMORY-RESIDENT	Feb	45	*May have problems using memory-resident programs with Framework*
	TEXT FRAME	Mar	46	*A containing frame can hold more than one 64K text frame*

Sample Programs

Product	Entry	Issue	Page	Description
dBASE II	CSEED2.PRG	Jan	17	*Program to create a seed value for random number generation*
	EVAL2.PRG	Jan	22	*Calculates the standard deviation and mean value of random numbers*
	HEC-DEC.CMD/PRG	Feb	18	*Converts hex program into a dBASE II POKE sequence*
	LEADZERO.PRG	Feb	20	*Program to replace leading blanks with zeros*
	MODSTUC.PRG	Jan	10	*Program to emulate MODIFY STRUCTURE*
	RANDOM2.PRG	Jan	18	*Program to generate a random number*
	WINNERS.PRG	Jan	20	*Program to select random winners from a database file*
dBASE III	CSEED3.PRG	Jan	19	*Program to create a seed value for random number generation*
	DECRYPT.PRG	Feb	33	*Data decryption program*
	ENCRYPT.PRG	Feb	31	*Data encryption program*
	EVAL3.PRG	Jan	23	*Calculates the standard deviation and mean value of random numbers*
	FILEPROC.PRG	Mar	22	*Procedure file to print program listings*
	PRGPRINT.PRG	Mar	21	*Program to display menu selection for print program listings*
	RANDOM3.PRG	Jan	20	*Program to generate a random number*
dBASE III PLUS	COLOR.PRG	Mar	12	*Program to SET COLOR from a menu in dBASE III PLUS*
	LISTHIST.PRG	Feb	8	*Program to LIST HISTORY to a file in dBASE III PLUS*
	VIEW.PRG	Feb	12	*Program to toggle between table view and form view*
Framework II	DATA TABLE	Feb	35	*Program to create data tables in Framework II*
	LABELPGM	Mar	14	*Program to automate mailing labels*
	LSTRING	Jan	25	*Function to extract the left portion of a character string*
	MID STRING	Jan	31	*Macro to extract the mid portion of a string in word frames*
	PFS TRANSFER	Mar	37	*Program to transfer files from pfs:file to Framework II*
	RSTRING	Jan	26	*Function to extract the right portion of a character string*
	RUNDBASE	Mar	27	*Running dBASE from within Framework II*
	STRING FUNCTIONS	Jan	24	*User-defined string functions used in spreadsheet and database frames*
	TABSTOP	Mar	18	*Macro for putting variable tab stops into word frames*
	TABSTRIP	Jan	33	*Program to remove tabs from a word frame*
Other	CHOOSE.COM	Mar	33	*An assembly language routine used to choose menu options with MENU.BAT*
	MENU.BAT	Mar	32	*DOS batch file is controlling program for front-end menu system*
	MENU.DAT	Mar	33	*DOS batch file, called from MENU.BAT, displays menu screen*

Miscellaneous

Subject	Issue	Page	Description
CORRIGENDA	Feb	27	Corrections to the March 1985 issue
	Jan Mar	36	Corrections to September 1985, November 1985, and December 1985 issues
		3	Corrections to the July 1985, October 1985, and December 1985 issues
ON-LINE SUPPORT	Jan	3	Information pertaining to CompuServe, The Source, and Onepoint
SUBSIDIARY OFFICES	Jan	56	List of Ashton-Tate international subsidiary offices
TRAINING CENTERS	Jan	54	Listing of Ashton-Tate Authorized Training Centers
	Mar	47	Listing of Ashton-Tate Authorized Training Centers

TechNotes
P.O. Box 3729
Escondido, CA 92025-0929

ADDRESS CORRECTION REQUESTED
RETURN POSTAGE GUARANTEED

BULK RATE
U.S. POSTAGE
PAID
PERMIT NO. 27
CULVER CITY, CA